A Neural Network for Tornado Prediction Based on Doppler Radar-derived Attributes

Caren Marzban[1,2,3]

and

Gregory J. Stumpf[1,2]

[1] National Severe Storms Laboratory, Norman, OK 73069

[2] Cooperative Institute for Mesoscale and Meteorological Studies, University of Oklahoma, Norman, OK 73019

[3] Department of Physics, University of Oklahoma, Norman, OK 73019

March 18, 1998

Abstract

The National Severe Storms Laboratory's (NSSL) Mesocyclone Detection Algorithm (MDA) is designed to search for patterns in Doppler velocity radar data which are associated with rotating updrafts in severe thunderstorms. These storm-scale circulations are typically precursors to tornados and severe weather in thunderstorms, yet not all circulations produce such phenomena.

A neural network has been designed to diagnose which circulations detected by the NSSL MDA yield tornados. The data used both for the training and the testing of the network is obtained from the NSSL MDA. In particular, 23 variables characterizing the circulations are selected to be used as the input nodes of a feed-forward neural network. The output of the network is chosen to be the existence/nonexistence of tornados, based on ground observations. It is shown that the network outperforms the rule-based algorithm existing in the MDA, as well as statistical techniques such as Discriminant Analysis and Logistic Regression. Additionally, a measure of confidence is provided in terms of probability functions.

1 Introduction

Recently, Neural Networks (NN) have begun to emerge as an entirely novel approach for the modeling of complex, nonlinear, dynamical phenomena (Hertz, Krogh, & Palmer, 1991; Masters, 1993; Müller & Reinhardt, 1991). The range of applications varies from predictions in the Stock Market (Collins, Ghosh, & Scofield, 1988), to the study of particle interactions in high-energy physics (Peterson & Rögnvaldsson, 1992; Kantowski & Marzban, 1995). The utility of NNs is most present in disciplines where intrinsic nonlinearities in the dynamics preclude the development of exactly-solvable models. In such cases, a trained neural network is synonymous with a solvable model. Although, qualitative, physical understanding may be lacking, highly accurate predictions can be made. In a field such as meteorology, all of these criteria are present, in that the dynamics is inherently nonlinear, and predictions comprise a central goal.

Neural networks have been previously employed in lightening prediction (Frankel, 1990) and in cloud classification/identification (Bankert, 1994 and Peak, 1994). Radar data has been used for neural network discrimination between ground clutter and point targets (Cornelius & Gagnon, 1993), as well as in microburst prediction (Keller, 1994), but a hinderance has often been the lack of data for training. However, the recent addition of the network of Doppler Radars to the paraphernalia of meteorologists is likely to make the shortage of data a problem of the past. As a result, the heterosis of neural networks and doppler radar promises to be a fruitful one, and has been one reason for undertaking the present study.

2 The Mesocyclone Detection Algorithm (MDA)

Algorithms have been designed to detect a variety of severe weather signatures, such as hail, high winds, and tornados in Doppler weather radar data. Mesocyclone detection algorithms are designed to detect the storm-scale circulations which are associated with rotating updrafts in thunderstorms, more commonly known as supercells. A mesocyclone detection algorithm resides on the National Weather Service's (NWS) Weather Surveillance Radar - 1988 Doppler (WSR-88D) system, and is used operationally as guidance to meteorologists to warn the general public of tornados and other severe weather associated with supercell thunderstorms.

The National Severe Storms Laboratory (NSSL) has been developing an enhanced Mesocyclone Detection Algorithm (MDA)[1] which contains a variety of more robust techniques for searching out the patterns within Doppler-radar velocity data which are associated with storm-scale circulations. Previous detection methods were constrained in that particular thresholds and rule bases were designed to detect only certain types and scales of circulations. Circulations were thresholded for dimension (such as depth and height of the base above ground), and for strength (such as rotational velocity). The NSSL MDA relaxes those constraints and is designed to detect a wide range of circulations of varying dimensions and strengths. The advantages of the new algorithm are two-fold. First, the algorithm is more robust than earlier versions, and detections are more accurately defined. Second, with the detection of additional circulations which may not meet specified "rules", statistical methods can be applied to determine the probability that particular circulations are associated with tornados or other forms of severe weather at the ground.

In the MDA there are three types of rules for mesocyclone classification: strength rules, dimension rules, and time associations. The former is based on the threshold values for velocity difference (m/s) and shear ($\times 10^{-2}/s$) for each located 2D circulation feature; these are 30 and 6, for ranges 0-100km, and decrease linearly to 75% and 50% of the original values between 100-200km, respectively, and maintain a threshold of 22.5 and 3 beyond 200km. The dimension rule requires a continuous depth of 2D features with the following conditions in a "NSSL 3D couplet": Half-beamwidth depth

[1] The details of the inner-workings of the MDA will be presented elsewhere (Stumpf, Marzban, Rasmussen, 1995).

Figure 1: A snap-shot of a WSR-88D radar data and a mesocyclone detection.

of 3D couplet must be at least 3km, the base of the 3d couplet must be below 3km, and no 2D features must be used above 10km. Finally, a "NSSL mesocyclone" is defined as any "NSSL 3D couplet" on a current volume scan which is time associated (using a distance check) with *any* 3D circulation (of any strength, including weak circulations) from a previous volume scan. Figure 1 is a snap-shot of a WSR-88D radar data and a mesocyclone detection. Displayed are also the associated outputs produced by Radar and Algorithm Display System (RADS; Sanger et al., 1995 and Eilts et al., 1996).

3 Neural Networks - A Review

There exists a plethora of statistical techniques for analyzing data, and in combination with the variations emerging due to specialized needs, one is faced with the difficulty of choosing the "best" method of analysis. At the same time, traditional methods invariably have inherent assumptions that limit their applicability. For instance, in Discriminant Analysis (DA) distributions are assumed to be gaussian (normal) - an assumption that may easily be violated. In Multiple Linear Regression the underlying relation is assumed to be linear in the parameters. In Polynomial Regression, the function is assumed to be some prespecified polynomial. In cases where the outcome is binary, the variant of such regression models is Logistic Regression, where the underlying map is assumed to be the logistic function (see below). All of these models incorporate assumptions whose violation may jeopardize the predictive capabilities of the model. Neural Networks, on the other hand, are extremely robust in regards to *a priori* distribution of the data. Furthermore, in a Neural Network analysis no assumptions are made regarding the underlying relations; by varying the number of hidden nodes (see below), one effectively parametrizes the space of all functions. For more details, see the Appendix, and for a precise statement of "space of all functions", see Hornik, Stinchcombe, and White (1989).

Neural Networks are designed to extract existing patterns from noisy data. The procedure involves training a network (training phase) with a large sample of representative data, after which one exposes the network to data not included in the training set (validation, or prediction phase) with the aim of predicting the new outcomes. Specifically, a feed-forward NN has some number of input and output nodes, characterizing, respectively, the independent and dependent variables of an underlying map which is to be learned by the network. There may also be one or more hidden layers with some number of nodes on each (see Figure 2). The number of hidden layers/nodes is a quantity that must be determined emprirically and for each separate situation. Typically, one experiments with a variety of architectures in order to find one that optimizes the performance of the network (on the validation set; see below). The value σ_j of the j-th node (on the first hidden layer) is given by $\sigma_j = f(\sum_i \omega_{ji}\tilde{\sigma}_i + \theta_j)$, where ω_{ji} are the weights connecting the i-th input node (whose value is $\tilde{\sigma}_i$) to the j-th hidden node whose activation threshold is θ_j. A similar rule applies to the nodes on the second hidden layer, as well as those on the output layer, with the values of the nodes on any layer being determined from the ones on the previous layer. Typically, the activation function f for the first layer is a sigmoid function, and for any remaining layer is either a sigmoid or a linear function. In this application, the activation function for all the layers is taken to be the logistic (or fermi) function,

$$f(x) = \frac{1}{1 + \exp(-x)}.$$

The choice of a logistic activation function does not comprise an assumption regrading the underlying function. It can be shown (Hornik, et al., 1989) that the performance of the network is insensitive to the choice of the activation function. All that is required for learning (i.e. convergence) to occur is for the activation function to be smooth and bounded. The particular choice of a logistic function is based on a property that makes it numerically economical, namely that its derivatives can be calculated from the function itself, i.e. $f'(x) = f(f-1)$.

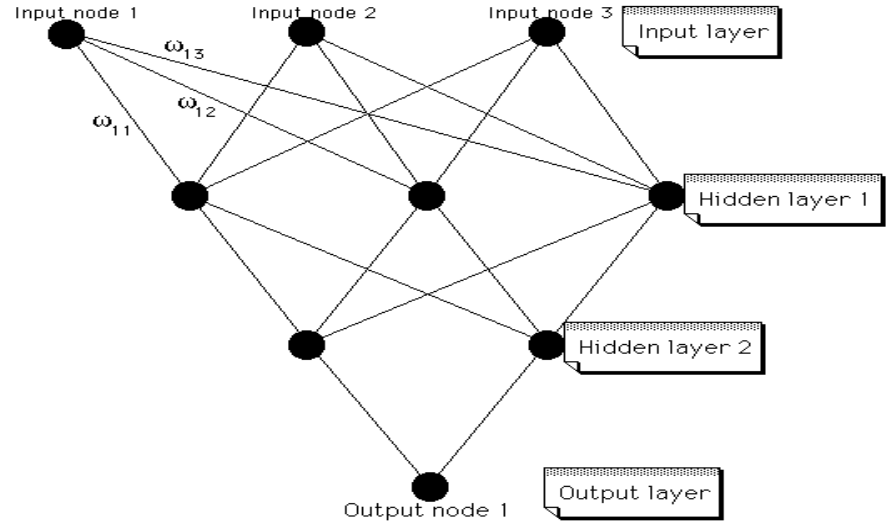

Figure 1: A Feed-forward Neural Network with 3 input nodes and 1 output node, with 5 hidden nodes on 2 hidden layers. Also shown, are 3 of the 17 weights (links).

Training a NN involves minimizing the Mean Square Error of the outputs of the network

$$MSE(\omega) = \frac{1}{N}\sum_{i=0}^{N}(a_i - p_i(\omega))^2,$$

where the ω are the weights (parameters) of the network (Figure 2), N is the number of cases in the training set, and a_i and p_i are the actual and the predicted values of a single output node. For larger number of output nodes, one introduces another $\sum$ into the equation that ranges over the number of output nodes. The set of ω's that minimize E are the crucial quantities, and when they are found the network is said to have been trained. Sometimes the values of the ω's can be found analytically, and exactly. In most practical cases, however, when nothing analytic is known about the function that is to be learned, an iterative approach has to be employed; the short-coming of this unfortunate state of affairs is that all such iterative methods are insensitive to whether the found minimum of E is a local or a global one. A network that is caught in a local minimum is said to be a confused network (Müller & Reinhardt, 1991). In order to avoid, or escape, local minima a variety of techniques have been developed, such as Genetic Algorithms and Simulated Annealing (Masters, 1993).

The initial randomly assigned weights are modified according to some learning algorithm (below) in order to minimize this quantity (also called the energy function). The updating of the weights can be done either after the presentation of each member of the training set, or after all the cases in the training set have been presented. In the present application, the latter is employed. Subsequently, the weights are "frozen", the input nodes are assigned values (not in the training set), and the value of the output nodes, as determined from the trained weights leading to them, are taken to be the predicted value of the dependent variables in question. The performance of the network is monitored by a validation set, which is another set of known independent/dependent variables, not used in the training, and whose target values are compared to the values predicted by the trained network. It is the performance of the network on this validation set that is the true measure of the predictive capability of the network.

4 The Method

The circulation data base used for training the neural network was "truthed" to determine which circulations were associated with reports of actual tornado events at the ground. Any circulation detected on a particular volume scan of radar data (the sampling rate of a radar volume scan is approximately 6 minutes) can be associated with a report of a tornado, hail greater than 1.9 cm in diameter, and/or winds in excess of 25 m/s. These three occurrences all classify a circulation as being "severe", and the occurrence of a tornado (either alone or in combination with other severe weather phenomena) renders the circulation "tornadic". If a circulation is detected within 20 minutes prior to a ground report of severe weather or a tornado, or 5 minutes after a report, the circulation is classified as a "prediction" of severe weather or a tornado (depending on the ground report). The neural network is then trained to determine whether a circulation will produce a tornado within the next 20 minutes, a suitable "lead-time" for advanced severe weather warnings by the NWS.

The particular NN program used in this study is a modified version of one obtained from Masters (1993). The original source codes, written in C++, were designed to be compiled and executed on DOS machines. For our purposes the source codes were modified to run in the UNIX environment. The modified version allows us to use a large number of nodes on each layer as well as to view the network's weights. Usually the weights are uninterpretable due to the presense of hidden layers and the nonlinearity of the activation function. Even with no hidden layers, linear correlations in the input data (nodes) can render the weights uninterpretable. However, with proper care one can still gain nontrivial information from the weights (see Appendix).

Also, the statistical method of Discriminant Analysis (see Appendix) is performed to provide a comparison with the results of the NN. Logistic Regression (i.e. NN with no hidden nodes - see Appendix) also provides for an additional comparison. The NN, Discriminant Analysis, and Logistic Regression will all be compared to the rule-based algorithm present in the MDA.

The mesocyclone attributes that were selected to be used as the input nodes of the network are given below. These 23 variables are believed to play an important role in determining whether or not a given circulation is tornadic. They are outputs of the WSR-88D and NSSL mesocyclone algorithms; they are also traditionally used by the National Weather Service to diagnose mesocyclones during severe storm warning operations. Currently, the MDA does not compute additional variables. However, in the future, more inputs derived from radar reflectivity and near-storm environmental data are to be incorporated into the NN.

It is possible to reduce the number of input nodes by employing only the best predictors (as determined from some other statistical method such as Step-wise Discriminant Analysis), or a combination of some of the variables (as found from Principle Component Analysis). However, by experimenting with smaller networks (i.e. 8, 10, 14, ... inputs) we have verified that the inclusion of all 23 variables does not hinder the performance of the network, and as a result the entire set is included.

1. Base (m)
2. Depth (m)
3. "Strength rank" (0-9)
4. Low-altitude diameter (m)
5. Maximum diameter (m)
6. Height of maximum diameter (m)
7. Low-altitude rotational velocity (m/s)
8. Maximum rotational velocity (m/s)
9. Height of maximum rotational velocity (m)
10. Low-altitude shear (m/s/km)
11. Maximum shear (m/s/km)
12. Height of maximum shear (m)

13. Low-altitude gate-to-gate velocity difference (m/s)
14. Maximum gate-to-gate velocity difference (m/s)
15. Height of maximum gate-to-gate velocity difference (m)
16. Core base (m)
17. Core depth (m)
18. Age (s)
19. Strength index (MSI) weighted by avg density of integrated layer
20. Strength index (MSIr) "rank"
21. Relative depth (%)
22. Low-altitude convergence (m/s)
23. Mid-altitude convergence (m/s)

These attributes are all based on Doppler velocity data; the inclusion of reflectivity data and/or near-storm environmental (sounding) data is currently under consideration.

As mentioned above, the energy surface whose minimum is to be found is well-known for being infested with local minima. For this study we employed Simulated Annealing (see Appendix), both for initiating a set of weights that could then be evolved according to the learning algorithm, and for attempting to escape the local minimum when the learning algorithm was incapable of doing so. The particular learning algorithm adopted here was Conjugate Gradient, a variation of the more familiar back-propagation method (Masters 93).

In training an NN some transformation of the data is inevitable. For instance, given the range of the fermi function, one must scale the target values to lie in the range 0 to +1 (for numerical reasons, it is advisable to shrink that range to 0.1 to 0.9). Because of the asymptotic behavior of the logistic function for both small and large values of the domain, it is beneficial to scale the independent variables to lie in a similar range as well.

For the prediction and detection of 2 classes, as in tornado versus non-tornado (henceforth, generally referred to as "0" or "1"), whereas one output node would suffice, two output nodes were present. During the training phase, the presence of a tornado is indicated by a (0,1) for the two output nodes, respectively, while a (1,0) indicates the absence of such phenomena. During the validation phase, the values of each output node are Real numbers ranging from 0 to 1. In this phase, the classification of a circulation is based on the larger of the 2 output nodes. It is also possible to transform the outputs of the NN into probabilities for the respective events. This will be treated, in detail, below.

In a NN application both the architecture and the training data itself must often be manipulated in order to optimize the prediction capability of the network. The optimal architecture is arrived at by selecting the number of hidden nodes, from a variety of attempts with different number of hidden nodes, that optimize the predictions. That an optimal network can be found by considering different number of hidden nodes can be seen as follows: Hidden nodes are responsible for introducing more parameters (weights) that can then be modified by the learning algorithm. Having no hidden nodes is equivalent to performing a "linear" fit to the data, while having an unreasonably large number (say, a million!) of hidden nodes can lead to a fit curve that goes through every, and all, of the data points (i.e. over-fitting). If the underlying relations are nonlinear, or if there is "noise" in the data, it is clear that both of these limiting networks will have no predictive capabilities whatsoever; this breakdown would be due to nonlinearities in the data, in the former case, and due to the learned function being driven by random noise, in the latter case. As a result, by monitoring the performance of a sequence of networks with an increasing number of hidden nodes, one can identify the one with the highest predictive power.

Manipulations of the training set involve varying the ratio of the number of 0's (non-events) to 1's (events), and again selecting the ratio that optimizes the predictions. The corresponding ratio for the validation set is not a parameter to be varied, for it is to be representative of the ratio of the events in Nature. In particular, the ratio of 1's to 0's in the entire data set is 7.7%, and so the

same ratio is adopted for the validation set.

The performance of the network is gauged in terms of the Critical Success Index (CSI) (Donaldson, Dyer and Krauss, 1975) of the validation set. It is derived from the validation contingency table (otherwise known as the "Confusion Matrix"), or in short the C-matrix,

$$\text{C-matrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} \text{0's predicted as 0's} & \text{0's predicted as 1's} \\ \text{1's predicted as 0's} & \text{1's predicted as 1's} \end{pmatrix} = \begin{pmatrix} \cdot & \text{false alarms} \\ \text{misses} & \text{hits} \end{pmatrix},$$

as

$$\text{CSI} = \frac{d}{b+c+d}.$$

The Probability Of Detection (POD) and the False Alarm Rate (FAR) are also calculable from the C-matrix:

$$\text{POD} = \frac{d}{c+d}, \quad \text{FAR} = \frac{b}{b+d}.$$

The use of CSI as a measure of performance needs some justification. The CSI has several defects: Most notably, it does not take into account the correct prediction of nonevents (element a in the C-matrix), and it has also been termed "inequitable" by Gandin and Murphy (1992). The latter authors suggest the use of a score originally defined by Pierce, and known today as Kuipers' index or also as the True Skill Score. However, Doswell, et al. (1990) have shown that for rare-event forecasting the True Skill Score approaches POD, ignoring FAR altogether; their recommended index is the Heidke Skill Score. Although CSI is the measure that is employed throughout this article, we have calculated the Heidke Skill Scores as well and have found results that duplicate those presented here. The choice of the CSI was based on the requirements of the users of this neural network, and its similarities with the Heidke Skill Score (in this particular situation) entirely justify its use. Such issues are considered in further depth in (Marzban & Stumpf, 1995).

It is important to point out that, henceforth, all of the reported scores, CSI, POD, and FAR are those of the *validation* C-matrix, and hence are faithful measures of the *predictive* performance of the neural network.

5 Confidence Measure: Probabilities

As it is, the network can predict in a "yes" or "no" mode, and its prediction efficiency can be gauged in terms of the CSI of the validation set. However, after the optimal network is found, it is possible to transform the output of the network to a probability, reflecting the level of confidence associated with a given outcome. To that end, the two output nodes are functionally combined into one fictitious output node, σ. The "mixing function" in this application was taken to be the logistic function:

$$\sigma = f(\beta(\sigma_1 - \sigma_2)),$$

where f is the logistic function (defined above), σ_1, σ_2 are the two output nodes, and β is a parameter that measures the strength of the mixing; we have found that the ultimate results are insensitive to the exact value of the β parameter.

There exist two conventional techniques for estimating confidences. The first is based on hypothesis testing and deals with the probability of an observation, given that it belongs to the null hypothesis. The second is based on Baye's inference and deals with the probability of belonging to a hypothesis given an observation. In the former, the necessary quantity is the distribution of the activations of the output node of network under the null hypothesis. In this way, one arrives at a

probability of an observation belonging to the alternative hypothesis. As a result, nothing is said about the probability that the observation belongs to the null hypothesis itself. That would require a knowledge of the distribution of the activations of the output node under the alternative hypothesis. This often leads to the situation where the two probabilities do not sum to 1. Not independently of this problem is the inadequacy of hypothesis testing in the classification into more than 2 classes.

The second method, although involving more assumptions, does not suffer from these shortcomings. As a result, in this article we appeal to Baye's theorem to provide the measure of confidence. Specifically, the *posterior* probability that the tornado hypothesis was in effect when a variable σ was measured, can be calculated as

$$P_1(\sigma) = \frac{p_1 L_1(\sigma)}{p_0 L_0(\sigma) + p_1 L_1(\sigma)},$$

where p_0, p_1 are the *prior* probabilities, and L_0, L_1 are the likelihood functions for the random variable σ to be, respectively, a tornado or a non-tornado. In this formalism, the probability of the nontornado hypothesis to have been in effect when variable σ was sampled is simply $P_0(\sigma) = 1 - P_1(\sigma)$. We estimated the prior probabilities p_0, p_1 with the fractions $N_0/(N_0+N_1)$ and $N_1/(N_0+N_1)$, respectively, where N_0 and N_1 are the corresponding sample sizes.

An estimate of the likelihood functions has been proposed by (Parzen, 1962):

$$L(\sigma) = \frac{1}{n\lambda} \sum_i W(\frac{\sigma - \sigma_i}{\lambda}),$$

where σ_i are a random sample of size n, and the W is a weighting function. Whereas Parzen's estimator asymptotically approaches the true density function as the sample size increases, for a broad range of W's, a common choice is the gaussian function

$$W = \exp^{-[(\sigma - \sigma_i)/\lambda]^2},$$

which we shall adopt. It is important to note that such assumptions affect only the probabilities derived from the NN, and not the NN itself. Parzen (1962) also discusses the insensitivity of even the final probabilities to such choices. Here, the random sample σ is drawn from the training data. The parameter λ is a "smoothing parameter" that is to be fixed; the final results are insensitive to the specific value of λ. Figure 3a shows an example of the distribution of the (fictitious) output activation for the "0"'s and the "1"'s, and Figure 3b shows the corresponding likelihood functions as derived from Parzen's estimator at $\lambda = 0.1$.

6 Results and Conclusions

In what follows, the tornado prediction results will be presented in terms of the CSI scores, and probabilities; an analysis of severe weather data, as well as considerations of alternative scoring methods, such as the Heidke Skill Score or the True Skill Statistics (e.g. Doswell, Davies-Jones, Keller, 1990) will be presented elsewhere (Marzban & Stumpf, 1995).

The training and the validation sets were selected from a total of 3,258 circulations detected by the MDA. The number of tornadic circulations was 235, and the remaining 3,023 were nontornadic, making for a ratio of 235/3023 = 0.078. The 235 "1"'s were divided into two groups of 155 and 80 cases to be used in the training set and the validation set, respectively. The ratio, 0.077, was employed to select 1033 nontornadic circulations in the validation set. The number of nontornadic cases in the training set was allowed to vary from 100 to 1,990 (= 3023 - 1033). In short,

$$\begin{pmatrix} \text{no. of 0's in training} & \text{no. of 0's in validation} \\ \text{no. of 1's in training} & \text{no. of 1's in validation} \end{pmatrix} = \begin{pmatrix} \text{variable} & 1033 \\ 155 & 80 \end{pmatrix}.$$

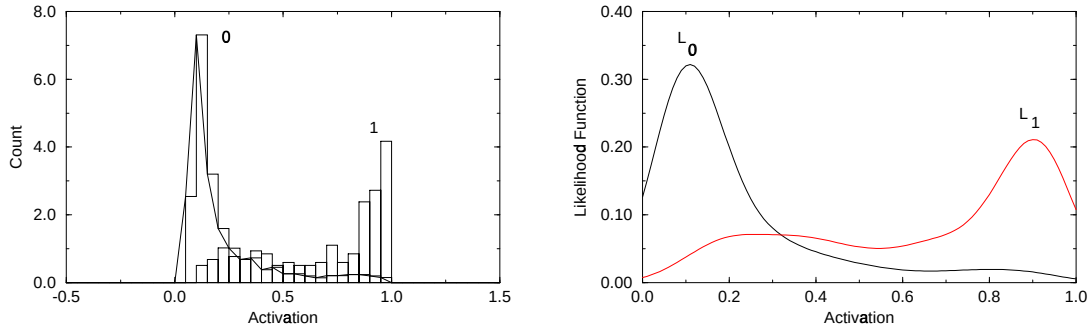

Figure 2: a) The distribution of activations, and b) Parzen's likelihood function ($\lambda = 0.1$).

To determine the optimal number of 0's in training, 100 were selected for training and the NN was tested on the validation set. The number of 0's in the training set was then enlarged in increments of 200, and the NN was tested on the validation set each time.

In order to assure that the selection of the circulations, either for training or validation, was not biased, the same procedure was repeated for 10 different random sets . The validation CSIs were, then, averaged over the different random sets. It is important to note that both the training set and the validation set were randomly selected, and so each of the 10 attempts represents a totally independent sampling of the data. Although the sampling may be independent, it is not true that the samples themselves are independent, since there is a great deal of overlap between them [1].

Figure 4 shows the seed-averaged validation CSI's as a function of the size of the training set, for networks with 0, 2, and 4 hidden nodes on one hidden layer. Evidently, the optimal architecture is one with 2 hidden nodes, and the optimal number of 0's in the training set is 900. The average CSI at this point is 34.3%.

One may wonder if the NN is outperforming other statistical techniques. Table 1 shows the validation CSIs for the 10 different training and validation sets, for 3 different algorithms: the rule-based technique currently in use in the MDA, Discriminant Analysis, and the Neural Network. The average CSI along with the 95% Confidence Intervals (CI) of the three methods are also provided. The absence of an overlap between the CIs of the three methods indicates that the NN is outperforming both MDA's rule-based (expert) system and Discriminant Analysis, when the performance is measured is terms of CSI. We have examined other measures as well (e.g. True Skill Score, Heidke Skill Score, etc.), and have found that the network still outperforms the other methods, albeit to different degrees for different measures. The matter of skill scores is treated elsewhere (Marzban and Stumpf, 1995).

The reasons why the NN is outperforming the other algorithms can be traced back to the origins of the 3 algorithms. MDA employs a rule-based algorithm; not only the rules may be wrong, but also they may not be covering every possible contingency. This static nature is a general problem with all rule-based algorithms. Discriminant analysis is a classification method based on several assumptions (see Appendix), namely the normality and the homoelasticity (i.e. the equality of the

[1] We thank Mike Eilts for pointing out this source of possible bias.

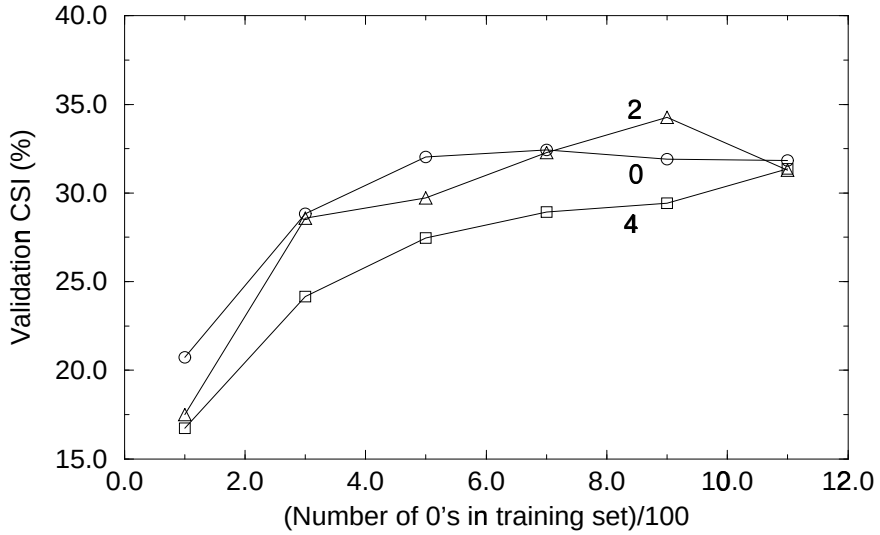

Figure 3: The average validation CSI as a function of the number of "0"'s in the training set, for 3 different networks with 0, 2, and 4 hidden nodes on one hidden layer.

seed	$CSI_{MDA}(\%)$	$CSI_{DA}(\%)$	$CSI_{NN}(\%)$	$POD_{NN}(\%)$	$FAR_{NN}(\%)$	C-matrix$_{NN}$
1	26.9	31.0	36.9	51.2	43.1	$\begin{pmatrix} 1002 & 31 \\ 39 & 41 \end{pmatrix}$
2	24.0	29.2	35.7	50.0	44.4	$\begin{pmatrix} 1001 & 32 \\ 40 & 40 \end{pmatrix}$
3	24.7	28.1	38.3	55.0	44.3	$\begin{pmatrix} 998 & 35 \\ 36 & 44 \end{pmatrix}$
4	28.7	27.7	33.6	58.8	56.1	$\begin{pmatrix} 973 & 60 \\ 33 & 47 \end{pmatrix}$
5	27.4	30.0	34.2	50.0	48.1	$\begin{pmatrix} 996 & 37 \\ 40 & 40 \end{pmatrix}$
6	28.0	28.8	32.5	47.5	49.3	$\begin{pmatrix} 996 & 37 \\ 42 & 38 \end{pmatrix}$
7	29.9	26.1	33.1	52.5	52.8	$\begin{pmatrix} 986 & 47 \\ 38 & 42 \end{pmatrix}$
8	21.3	28.7	29.1	46.2	56.0	$\begin{pmatrix} 986 & 47 \\ 43 & 37 \end{pmatrix}$
9	27.7	30.6	37.8	60.0	49.5	$\begin{pmatrix} 986 & 47 \\ 32 & 48 \end{pmatrix}$
10	21.5	26.5	31.7	47.5	51.3	$\begin{pmatrix} 993 & 40 \\ 42 & 38 \end{pmatrix}$
Avg.	26.0	28.7	34.3	51.9	49.5	.
95% CI	27.5/24.5	29.5/27.9	35.7/32.9	.	.	.

Table 1: The validation CSIs for MDA's expert system, Discriminant Analysis, and the Neural Network for 10 different training/validation sets (randomly selected from 10 different "seeds"). The POD and the FAR, as well as the validation C-matrices of the Neural Network are also shown. Using the 10 values for CSI, the 95% Confidence Intervals (CI) for the three methods are also calculated.

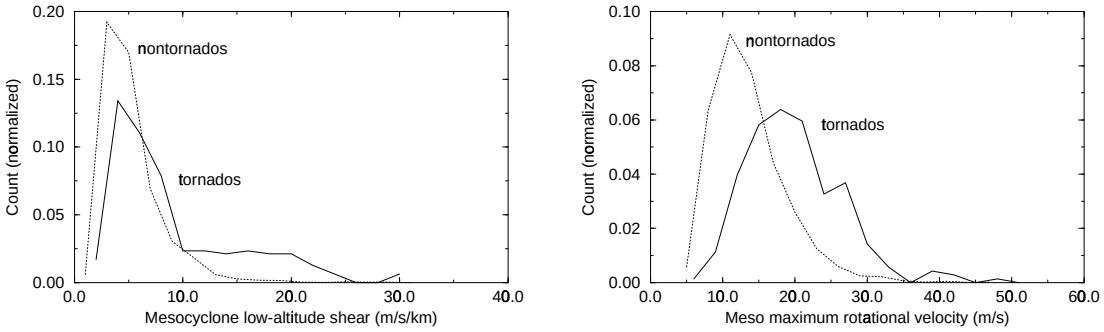

Figure 4: The distributions of two variables, for tornados and nontornados, exhibiting deviations from normality.

covariance matrices of the various classes) of the distributions. Indeed, from Figure 5 it is evident that the normality assumption is violated. The NN, on the other hand, is not restricted by empirical rules and is very robust under violations of such assumptions.

Finally, it is possible to display the results of the network in terms of trend probabilities. A given circulation was traced from formation (0 minutes) to demise (245 minutes). By monitoring this event in 5-minute intervals, 49 individual circulations were retrieved and shown to the trained network. Figure 6 shows the result. The bottom plot is simply the truth value, with 0 and 20 corresponding to "no tornado detected", and "tornado detected", respectively, and 10 labeling the circulations that are correlated with a tornado detection -20 minutes to +5 minutes from the time of the observation (but not at the actual time of the tornado), i.e. "tornado predicted". The labels 0, 10, and 20 are completely arbitrary numbers that were selected for aesthetic reasons. The top curve (with circles) shows the neural network's probabilities for the corresponding circulations to be correlated with a tornado (again, in a -20 minute to +5 minute window). Evidently, the network assigns appropriate probabilities to 38 of the 49 circulations. The filled circles mark the circulations that were misclassified by the NN in this probabilistic scheme.

In summary, a Neural Network (NN) is devised for the identification of tornado-yielding mesocyclones. To that end, procedures have been developed for determining the size of the training set and the number of hidden nodes necessary for optimum performance. It is shown that the NN found in this way outperforms a rule-based algorithm, Discriminant Analysis, and Logistic Regression. It is also shown that probabilities can be extracted from the NN, as a measure of confidence in the NN predictions. The approach is currently being applied to various other situations as well (e.g. severe weather-yielding mesocyclones, and tornados not produced from mesocyclones). The system is currently in a testing phase as a component of Severe Storm Analysis Package (SSAP) developed by the NSSL.

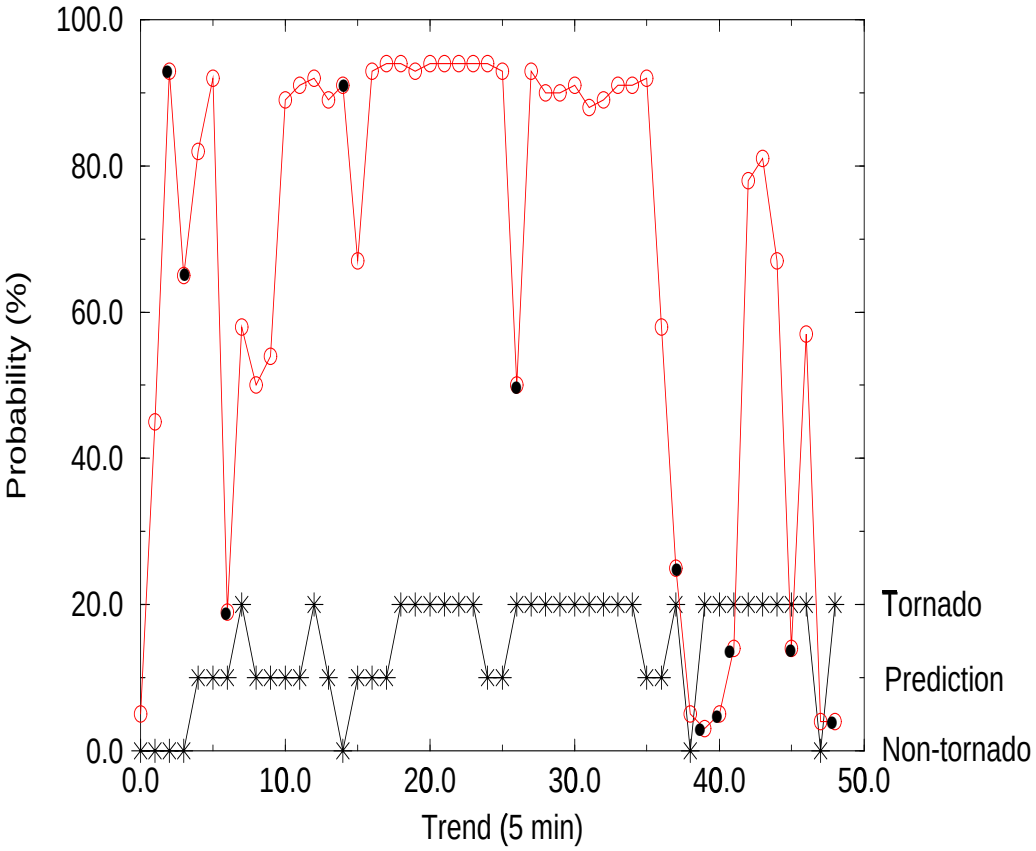

Figure 5: The probability of a tornadic circulation according to the neural network, and according to ground truth, for 49 circulations associated with a single supercell thunderstorm which produced several tornados.

7 Acknowledgments

We would like to thank Mike Eilts for his comments on the manuscript. Chuck Doswell is also acknowledged for a fruitful discussion on various skill scores.

8 Appendix

In this appendix we shall compare and contrast neural nets with some traditional statistical methods, discuss ways of interpreting the weights, and review Simulated Annealing as a method for both avoiding and escaping local minima.

The neural network methodology is inherently a statistical one, and as such can be compared with other more traditional methods. The simplest comparison has already been alluded to in the body of the article, namely that a neural network with a linear/logistic activation function and with no hidden layers is equivalent to Multiple Linear/Logistic Regression (MLR). A subset of MLR models is occupied by polynomial regression models; in these the underlying function is assumed to be a polynomial in the independent variables. This allows even linear models to fit a wide variety of functions, but the exact order of the polynomial for each of the independent variables has to be specified by the user, constituting an explicit assumption regrading the underlying function. The presense of hidden layers places neural nets in the realm of non-linear regression models but with one important difference - in the latter, the exact form of the nonlinear (in parameters) function that is to be fit to the data has to be assumed. In contrast, through the number of hidden nodes/layers the network parametrizes the space of all functions. In other words, by simply varying the number of hidden nodes/layers (up to, and excluding, that which overfits) one can systematically search and find the best fit, without making any explicit assumptions regarding the underlying function.

Yet another (traditional) method of discrimination is Discriminant Analysis. In this method, the probability density functions of the variables in each class $i = 1, 2, ...$ are assumed to be gaussian (normal):

$$P_i = e^{(\mathbf{x}-\mu_\mathbf{i})\mathbf{\Sigma_i^{-1}}(\mathbf{x}-\mu_\mathbf{i})},$$

centered around the respective class mean vectors $\mu_\mathbf{i}$, and with a covariance matrix $\mathbf{\Sigma_i}$, both of which are to be determined from a training set. Given the validity of this assumption, one can then classify an unknown case into the class with the highest P_i. Often, and in order to be able to attribute a weight to the variables involved, one makes the further assumption of homoelasticity (i.e. that $\mathbf{\Sigma_1} = \mathbf{\Sigma_2} = ...$). This gives rise to a linear equation in $\mathbf{x}$, with coefficients that can be interpreted as weights. If any of these assumptions are violated, the outcomes of the analysis may be inaccurate, or uninterpretable. Neural nets, by contrast, do not rely on such explicit assumptions regarding the underlying distributions or the covariance matrices. Of course, one can envisage pathological cases that "confuse" even the neural network, but research appears to indicate that even in such cases the network is less sensitive to such pathologies (Masters, 1993).

In Regression, Discriminant Analysis, and indeed all statistical methods, certain assumptions must be made to arrive at a ranking of the input variables in order of their strength in predicting the outcomes. Neural nets are no exception. Usually the goal of interpreting the weights is to identify the strongest predictors. This can be done even without analysing the weights; one can adopt a (forward) step-wise approach of examining networks with only one input node, two input nodes, etc., or the (backward) step-wise approach of including all the variables and systematically removing one input node, 2 input nodes, etc., until the relevant variables are isolated. This is the most brute-force approach and is thus very time-consuming. However, one can still extract such information from the weights, if the data is modified carefully. For instance, to assure that one input node does not obtain a large weight simply due to the large-magnitude values that it may be taking, it is wise to scale all of the input variables to lie in the same range. Also, since linear correlations between the input variables can cause the weights leading to them to become meaningless, one can either perform some nonlinear transformation on one of the linearly correlated variables, or to include only

the subset of input variables that are not linearly correlated. If one is interested in more than simply identifying the best predictors, and is more interested in the inner-workings of the network, then it is possible to process the data through a network that first identifies the best predictors (this being the nonlinear analog of principle component analysis) and then lead the output of this network into another that is devoted to prediction. The weights of the second (predicting) network will then be more transparent. These are but a few of the methods for revealing the inner-workings of neural nets, and the field is still evolving. Eberhart & Dobbins (1990), and Rumelhart & McClelland (1986) offer numerous examples of cases where the weights are interpretable.

Given the plethora of local minima in the error function (energy surface), it is important to appeal to some method of avoiding and/or escaping local minima. The method employed in this article has been Simulated Annealing (SA). SA is a large topic and manifests itself under various disguises, and as a result, we will not delve into its details. Instead, we will offer only an intuitive description using a mechanical paradigm. Imagine a mountainous landscape, consisting of great many local minima. Now imagine that there is a ball resting on this landscape which we would like to place in the deepest of these minima. It is a simple task to prove that the chances of succeeding in this task are maximized if the landscape is shaken, first violently, then less violently, followed by even gentler and gentler shakes. This process is referred to as the annealing process. In the neural net context, the initial weights are selected from a random distribution whose width is decreased systematically, analogous to the systematic decrease in the strength of shaking the box in the above example. In this way, one can obtain the best set of initial weights with the hope of avoiding the local minima. Of course, since the proof of SA's success is a probabilistic one, the method does not gaurantee success upon a single attempt. In that case, i.e. when the system has landed in a local minimum, one can use annealing again to find a better/deeper minimum. In this way, one expedites finding the global minimum, or at least a sufficiently deep local minimum.

8 References

Bankert, R.L., 1994: Cloud Classification of a AVHRR Imagery in MaritimeRegions Using a Probabilistic Neural Network. *J. Appl. Meteor.*, **33**, 909-918.

Collins, E., S. Ghosh, and S. Scofield, 1988: *Risk Analysis: DARPA Neural Network Study.* AFCEA International Press, 429 pp.

Cornelius, R., R. Gagnon, 1993: Artificial Neural Networks for Radar Data Feature Recognition. Preprints, *26th Conference on Radar Meteorology*, Norman, OK, Amer. Meteor. Soc. 340-342.

Donaldson, R. J., R. M. Dyer, and M. J. Krauss, 1975: An Objective Evaluator of Techniques for Predicting Severe Weather Events. Preprints, *9th Conference on Severe Local Stroms*, Norman, OK, Amer. Meteor. Soc., 321-326.

Doswell III, C.A., R. Davies-Jones, D. Keller, 1990: On Summary Measures of Skill in Rare Event Forecasting Based on Contingency Tables. *Weather and Forecasting*, **5**, 576-585.

Eberhart, R.C., and R.W. Dobbins, 1990: *Neural Network PC Tools; A Practical Guide.* Academic Press, 414 pp.

Eilts, M. D., J. T. Johnson, E. D. Mitchell, S. Sanger, G. Stumpf, A. Witt, K. W. Thomas, K. Hondl, D. Rhue, and M. H. Jain, 1996: Severe weather warning decision support system. Preprints, to appear in the *18th Conf. on Severe Local Storms*, San Francisco, CA, Amer. Meteor. Soc.

Frankel, D., I. Schiller, J. S. Draper, and A. A. Barnes, 1990: Investigation of the Prediction of Lightning Strikes Using Neural Networks. Preprints, *16th Conference on Severe Local Storms*, Kananaskis Provincial Park, Alberta, Canada, Amer. Meteor. Soc., 7-11.

Gandin, L. S., A. Murphy, 1992: Equitable Skill Scores for Categorical Forecasts, *Monthly Weather Review*, **120**, 361-370.

Hertz, J., A. Krogh, and R. G. Palmer, 1991: *Introduction to the Theory of Neural Computation.* Addison-Wesley, 414 pp.

Hornik, K., M. Stinchcombe, and H. White, 1989: Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, **4:2**, 251-257.

Kantowski, R., and C. Marzban, 1995: A Neural Network for Locating the Primary Vertex in a Pixel Detector. *Nuclear Instruments & Methods in Physics Research*, **A355**, 582-588.

Keller, D. 1994: Two Regression Fits to Microburst Data. Internal Report, National Severe Storms Laboratory, Norman, OK 73069.

Marzban, C., and G. Stumpf, 1995: A Neural Network for Severe Weather Prediction, and Measures of Skill. Preprint. National Severe Storms Laboratory, Norman, OK 73069 (submitted for publication).

Marzban, C., and R. Viswanathan, 1994: Stochastic Neural Networks with the Weighted Hebb Rule. *Physics Letters*,

Masters, T., 1993: *Practical Neural Network Recipes in C++.*, Academic Press, 493 pp.

Müller, B., and J. Reinhardt, 1991: *Neural Networks: An Introduction.* Springer-Verlag: The Physics of Neural Networks Series, 266 pp.

Parzen, E., 1962: On Estimation of a Probability Density Function and Mode. *Annals of Mathematical Statistics.* **33**, 1065-1076.

Peak, J.E., and P. M. Tag, 1994: Segmentation of Satellite Imagery Using Hierarchical Thresholding and Neural Networks. *J. Appl. Meteor.*, **33**, 605-616.

Rumelhart, D., J. McClelland, and the PDP Research Group, 1986: *Parallel Distributed Processing.* MIT Press, 415 pp.

Sanger, S. S., R. M. Steadham, J. M. Jarboe, R. E. Schlegel, and A. Sellakannu, 1995: Human factors contributions to the evolution of an interactive Doppler radar and weather detection algorithm and display system. Preprints, *11th Intl. Conf. on Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology*, Dallas, TX, Amer. Meteor. Soc., 1-6.

Stumpf, G., C. Marzban, and E. N. Rasmussen, 1995: The New NSSL Mesocyclone Detection Algorithm: A Paradigm Shift in the Understanding of Storm-Scale Circulation Detection. Submitted to the *27th Conference on Radar Meteorology*, Vail, CO, Amer. Meteor. Soc.