Neural Networks for Post-processing Model Output: ARPS

Caren Marzban[*]

Center for Analysis and Prediction of Storms
University of Oklahoma, Norman, OK 73019

and

Department of Statistics
University of Washington, Seattle, WA 98195

September 16, 2002

[*]http://www.nhn.ou.edu/~marzban

Abstract

The temperature forecasts of The Advanced Regional Prediction System are post-processed by a neural network. Specifically, 31 stations are considered, and for each a neural network is developed. The nine input variables to the neural network are: Forecast hour, model forecast temperature, relative humidity, wind direction and speed, mean sea level pressure, cloud cover, and precipitation rate and amount. The single dependent variable is observed temperature at a given station. It is shown that the model temperature forecasts are improved in terms of a variety of performance measures. An average of 40% reduction in mean-squared error across all stations is accompanied by an average reduction in bias and variance of 70% and 20%, respectively.

1 Introduction

Most numerical models of atmospheric processes employ some sort of statistical post-processing to improve performance. There are several ways by which this post-processing can take place. Two relatively common methods are referred to as Perfect-Prog and MOS (Wilks 1995). Although they both have specific pros and cons, the latter is better suited to some of the more contemporary approaches. In MOS, one typically derives a set of linear regression equations that relate the output of the model at some time to actual observations of the corresponding quantities (Glahn and Lowry 1972) at that time.

Recently, nonlinear methods have also been used for the post-processing (Casaioli et al. 2000). Neural networks constitute one such method (Masters 1993, Bishop 1996). Although the choice of a nonlinear method is not unique, neural networks are generally useful because in addition to being able to approximate a large class of functions, they are less inclined to overfit data than some other nonlinear methods. A brief introduction to the topic is provided in the next section.

The model under examination in this paper is the Advanced Regional Prediction System (ARPS). Details of the model can be found in Xue et al. (2000, 2001). The particular forecast quantity of interest is the surface temperature. Figure 1 shows that ARPS's temperature forecasts are biased, positively or negatively, depending on forecast hour. [1] As such, the model forecasts could benefit from some sort of

[1]The error-bars are standard errors. Assuming the errors are gaussian, then a 95% confidence interval would be twice as large as the standard errors. Note that during the first 10 forecast hours the bias=0 line overlaps with such confidence intervals. One could then argue that for those forecast

post-processing. In the present analysis surface observations (excluding temperature) at a given time are utilized to improve temperature forecasts at that same time. A time series analysis wherein observations at a given time are related to temperature forecasts at a later time will be considered in a separate article.

2 Neural Networks

Most discussions of Neural Networks (NNs) begin with a biological motivation. Although that is somewhat appropriate, there exists another motivation that receives less attention, namely the statistical one. The fact is that NNs are a generalization of traditional statistical methods for nonlinear regression and classification. Some may argue with the validity of the term "generalization" on the grounds that it suggests something above and beyond traditional methods. Many decades of research have shown that almost every traditional method can be represented with a NN, and vice versa (Bishop 1996, Masters 1993, Sarle 1994). The difference between NNs and traditional methods, if any, is in degree, not in kind.

For example, polynomial regression is capable of fitting, or in NN terminology learning, any polynomial. The space of polynomials is large enough to cover most practical applications. NNs, too, have universal approximation properties (Hornik et al 1989). As such, and certainly from a practical point of view, there is no qualitative difference in the capabilities of the two approaches. One quantitative difference is in the way the two handle the curse of dimensionality. That refers to the rate at

hours the forecasts are not biased at any statistically significant degree.

which the number of parameters (or weights) in a model grows with the number of independent variables. For example, the free parameters in a linear regression model are the regression coefficients, and their number grows linearly with the number of predictors. The same number in polynomial regression grows exponentially. NNs occupy a special midpoint because their number of free parameters grows linearly (as in linear regression), but NNs are still a full-fledged nonlinear model.

The rate at which the number of free parameters grows with the number of predictors is important because if that number is comparable (or larger) than the sample size selected for estimating those parameters, then the model can overfit the data. In NN terminology, the model memorizes the data, and thereby, has no generalization capability. For example, a fit that goes through every point in some data is driven by the noise in the data and will perform poorly on new data. Given that most applications are based on a finite amount of data, it is important to assure that the model is not overfitting. Because of the small number of free parameters, linear regression is the least likely of the three models to overfit, while it is incapable of fitting nonlinear data. At the other extreme, polynomial regression can fit highly nonlinear data, though it is also the most likely to overfit. The NN resides somewhere in between, because in spite of being capable of fitting any nonlinear data, overfitting is less of a concern. Of course, this is not to imply that NNs cannot overfit, for they certainly can.

Just as linear regression can be written as $y = \sum_i^{N_{in}} \omega_i x_i + \theta$, with N_{in} input (or independent) variables, the analogous equation for an NN is

$$y = g\left(\sum_{i=1}^{H} \omega_i \, f(\sum_{j=1}^{N_{in}} \omega_{ij} x_j - \theta_j) - \omega\right) \tag{1}$$

where H is called the number of hidden nodes, and $f(x)$ and $g(x)$ are some pre-specified functions. f and g are called activation functions, and although there are some commonly used choices (e.g., tanh(x), logistic(x)), their specific form is not too important. There are other variations of Eq. 1 that represent other types of NNs (Bishop 1996), but this one (the so-called one hidden layer, feedforward, supervised, multi-layered-perceptron) is the most commonly used. In this parametrization the analogy with linear regression is quite evident.

As with linear regression, the task is to estimate (or learn) the parameters - ω's and θ's - from data on x and y. However, there is also the quantity H which must be determined. One practice is to simply set it at some large number so that the NN has sufficient nonlinearity to represent the data. However, this can easily lead to overfitting.

The techniques for preventing NNs from overfitting are mostly the same as those designed for traditional methods. One method is called Bootstrapping (Efron and Tibshirani 1993). In its simplest form, one divides the data set into two sets - a training set and a validation set. The former is used for estimating the parameters by minimizing some error function (like mean square error). The resulting error is called the training error. The application of Eq. 2, with the parameters estimated from the training set, to the validation set produces the validation error. In bootstrapping, the partitioning of the data into training and validation is repeated many times, and the value of H that yields the lowest average validation error is taken to be the optimal value.[2]

As mentioned previously, NNs are not a quantum leap beyond traditional statistical methods. They can almost be considered a re-packaging of traditional methods. It just turns out that this particular packaging is very useful for many practical applications. The successes of NNs (i.e., Eq. 2) in the atmospheric sciences are numerous. Hsieh and Tang (1998), and Marzban (2002) offer a general introduction; some specific applications can be found in Marzban (2000, 1998), and Marzban and Witt (2000).

3 Data

Although ARPS produces forecasts at every grid point, the forecasts at the grid nearest a station are utilized for developing an NN for that station. [3] In what follows the various stations will be referenced by numerical labels.

ARPS produces 1hr forecasts up to 60hr in the future, starting at 0000 UTC. This forecast time is an important predictor, for it is known that the bias of the ARPS forecasts has a strong dependence on forecast time (Figure 1). For each station, model data consisting of hourly observations, are collected on the following variables: 1. Forecast hour, 2. model (i.e., ARPS temperature) forecast, 3. relative humidity,

[2]In addition to H, the magnitude of the weights also controls the nonlinearity of NNs. There exist numerous ways for controlling the magnitude of the weights, one of which is the weight-decay method (Bishop 1996, Sarle 1995).

[3]Grid size is 30 km with 50 levels.

4. wind direction, 5. wind speed, 6. mean sea level pressure, 7. cloud cover, 8. precipitation rate, and 9. precipitation amount. These variables are utilized for generating input to the NN. All of these are model variables (i.e., ARPS forecasts). The target variable (i.e., the variable the NN is supposed to forecast) is taken to be the observed surface temperature at the given station.

The linear correlation coefficient r between the 9 model variables and the observed temperature is given in Figure 2. Note that this figure is specific to one station (KSAC). It can be seen that the best predictor of observed surface temperature is the model forecast temperature itself, followed by relative humidity, and mean sea level pressure. Some of the worse predictors are the forecast hour, and precipitation rate. Wind speed is also a weak predictor. Although, this figure does provide some information regarding the predictive strength of the various variables, it is important to note that a low r value does not imply low predictive strength in a general sense, because the underlying relation may simply be nonlinear (Marzban 1999).

Table 1 shows the r values between each of the 9 variables. In other words, it is the correlation matrix. The absolute magnitude of the r's ranges from 0.01 to 0.72. The latter is the r between model forecast temperature and relative humidity. If a pair of variables had turned out to be very strongly correlated (say, $r = 0.95$), then one could have excluded one member of the pair. The ensuing reduction in the number of input variables would then render the statistical model less likely to overfit data. However, although $r = 0.72$ is relatively large, it is not sufficiently large to warrant the exclusion from the analysis of one member of the pair. [4]

The distributions of the variables (not shown) are all bell-shaped, with some slightly skewed. The exception is cloud cover (given in percent) whose distribution is relatively uniform between 0 and 100. As such, the variables can be standardized in a straight forward way (below).

4 Method

Typically, some amount of pre-processing is necessary before data can be presented to a NN (Marzban 2002). In the current analysis missing data are replaced with sample averages.[5] Wind speed and direction are transformed into their orthogonal components. The resulting variables are standardized to have a mean of 0 and a standard deviation of 1. This is done by subtracting the mean of each variable from each data point and dividing the result by the standard deviation of that variable (i.e., data are transformed into z-scores); this transformation is appropriate for variables with bell-shaped distributions. [6] The distribution of each of the variables is examined, and outliers are visually identified and removed. Given that many of the resulting variables are linearly correlated, a principal component analysis is performed on all

[4]The question "How large is large enough?" is difficult to answer in generality. The answer is problem dependent. However, it would be difficult to argue that two variables with $r = 0.95$ are not statistically equivalent. At the same time, an r of 0.70 would be difficult to consider as large enough to consider the two variables statistically equivalent.

[5]Replacing missing data with a sample average is one of the simpler methods of handling missing data. Equally simple, is the exclusion of cases with missing data. The latter is more appropriate if data is abundant, which is not the case in the present application.

[6]Temperatures are standardized differently. See, below.

the variables (excluding forecast time and model forecast temperature). [7] The data are then projected onto the principal components (to create principal scores), and standardized again for presentation to the NN. The data are finally divided into training and validation sets. This partitioning is performed several times in order to generate a number of different data sets, called bootstrap trial samples (Marzban 2002, Efron and Tibshirani 1993). [8]

One final pre-processing step is required, this one specific to the nature of the analysis. At the time of the development of the NNs, only a few months of model data were available: Feb. 14 to June 6, 2002. Of course, given the hourly nature of the forecasts, that duration is adequate for training an NN of moderate size, without fear of drastic overfitting. However, operationally, the NNs are utilized some number of weeks *after* their development is complete (and before a new set of NNs is developed). As such, the NNs are used during a time of the year from which no data were incorporated in the training set. Given that surface temperature has a strong seasonal dependence, this creates a slight problem. For instance, when the NNs are used during September, the typical temperatures (both observed and from the model) are in an entirely different range than that present in the training set.

The correct method of handling this problem is to develop season-specific NNs; however, that requires several *years* of data which are currently not available. An

[7]Forecast time is excluded because its dependence on all of the variables is nonlinear (in fact, periodic). As such, it would simply add "noise" to the principal component analysis. Model forecast temperature is excluded because it is highly correlated with all of the variables. It's inclusion in principal component analysis would simply overwhelm the remaining variables, and it would lead to the first principal component being model forecast temperature itself.

[8]The partitioning is performed on model runs, not the hourly forecasts.

alternative solution is to develop the NNs such that they are invariant under temperature transformations. To that end, all temperatures (model and observed) are "standardized" with *climatological* (not sample) means and standard deviations. Although available model data span only a few months, surface observations of temperature are available for many years, and so, reasonable monthly climatological estimates can be obtained. In short, temperature values $T(z, m)$, at forecast time z and month m are scaled as

$$T(z, m) \rightarrow \frac{T - \mu(z, m)}{\sigma(z, m)}, \tag{2}$$

where $\mu(z, m)$ and $\sigma(z, m)$ are the climatological mean and standard deviation, respectively. This transformation assures that the NNs are always exposed to a range of temperature values present in the training set, regardless of the time of the year they are utilized.

The optimal number of input nodes is determined by bootstrapping. In other words, the number of input nodes is systematically increased, beginning at 1. As a result, the number of free parameters (NN weights) increases. This increase in the number of parameters eventually leads the NN to overfit the training data. The point at which this overfitting occurs is where the average (over bootstrap samples) validation error begins to increase, while the average training error continues to decrease.

Specifically, first, NNs are developed with only 1 input (encoded among 4 nodes) [9]. Then, the model forecast temperature is added in as a second input. This is

[9]Four 3-valued nodes are utilized to represent forecast time. For example, a forecast at 0000 UTC is encoded by 4 input nodes taking values (-1,-1,-1,-1); a forecast at 0100 UTC is represented by (-1, -1, -1, 0), and 0200 UTC is encoded as (-1,-1,-1,+1). This coding scheme allows for forecast hours of up to 80hr, represented by (+1,+1,+1,+1).

followed by the development of NNs that also include the first principal scores of the data, and then the second principal scores, etc. The optimal number of input nodes is selected as that which yields the lowest validation error averaged over bootstrap trials. The optimal number of hidden nodes is found in the same way as the optimal number of input nodes. The single output node represents the observed temperature for a given station at a given forecast time.

Given that the problem at hand is regression-type (as opposed to classification-type), the choice of the error function to be minimized during the training phase is somewhat ambiguous (Marzban 2002). We will assume that the underlying errors are gaussian (normal), and later confirm that the assumption is valid. Given that assumption, the natural error function is the Mean Squared Error (MSE).[10]

Finally, it must be said that the activation function of the hidden layer is taken to be the logistic function, while that of the output node is a linear function. The former assures that the NN is nonlinear, and the latter allows for output values in the range of the Real numbers. The training algorithm is conjugate gradient (Masters 1993), and Bayesian methods (MacKay 1996, Marzban 1998) are employed to determine the magnitude of the weights. The minimization of any nonlinear error function is apt to involve local minima. This problem is handled by re-initializing the training phase from different initial weights several times; the deepest local minimum is taken to be the global minimum.

[10]Given a gaussian distribution of errors, it can be shown that parameter estimates based on the minimization of MSE are equal to maximum likelihood estimates (Wilks 1995, p. 108-110).

5 Results

The optimal number of input nodes is found to be 7 (4 to encode forecast time, 1 to represent model forecast temperature, and 2 for the first two principal scores). In other words, it was found that in addition to forecast time and model forecast only the first two principal components of the data are sufficient for making optimal forecasts. Note that, in general, the amount of variance explained by each principal component is quite independent of the predictive strength of that principal component. As such, there is no guarantee that the first principal component should be a useful predictor at all. In the present case it simply turns out that it is.

The optimal number of hidden nodes - one of the measures of the nonlinearity of the underlying function - is found to vary with station. Some stations require as few as 2 hidden nodes, while others call for as many as 8. One station requires 0 hidden nodes, suggesting that the underlying relation between model and NN forecasts is linear. Recall that the optimal number of hidden nodes is that which yields the lowest average (over bootstrap trials) validation error.

Note that the above procedure for arriving at the optimal number of hidden nodes also constitutes a comparison between NN and traditional model output statistics (MOS), at least implicitly. MOS is based on linear regression, and an NN with zero hidden nodes is equivalent to linear regression. As such, when the optimal number of hidden nodes is nonzero - which is the case for all the stations examined here - the NN outperforms MOS.

Figure 3a shows the scatterplot of the observed vs. NN forecast temperatures at one station. The top (bottom) panel is for the training (validation) set. The absence of a nonlinear pattern implies that the NN has learned the correct underlying function. The linear correlation coefficient is $r = 0.94$ for both the training and the validation set. The diagonal line on the scatterplot represents perfect performance. The data are randomly distributed about this line, as they should be. The slope of a least-square fit through the data can suggest possible problems. However, in this case the slopes are nearly 1.0, indicating a good NN fit. By contrast the scatterplot for the model forecast temperatures (Fig. 3b) displays a nonlinear pattern, suggesting that the model forecast temperatures have a nonlinear relationship with observations. The linear correlation coefficient ($r = 0.92$) is slightly lower than that of the NN, but the slope of the least-square fit (0.8) is considerably lower than that of the NN. Note the deviation from the diagonal (perfect performance) line.

In addition to scatterplots, residual plots also provide a useful means for assessing performance. A residual is defined as the difference between forecasts and observations (former minus latter), and a residual plot is the plot of the residuals vs. the forecasts. Fig. 4 is the residual plot for the NN and model forecast temperatures. Ideally, one would expect to see a random scatter of points around the residual=0 line. Although the NN displays such a pattern (Fig 4a), the model itself displays a clear nonlinear pattern (Fig 4b), again suggesting that the model is not producing optimal forecasts. Various attributes of these diagrams could be quantified, but the figures themselves carry more information than any finite list of scalar measures. For this reason, no further analysis of the residuals is performed. Incidentally, the curious "tails" in each of the figures is a consequence of the replacement of missing data with sample means.

Figure 5 shows the bias and variance of the forecasts as a function of forecast time. The top (bottom) figure is for the training (validation) set. The first feature that stands out is the overall increase in variance of the model forecast temperatures with longer forecast times (the curve labeled var_model). The NN reduces this variance at all forecast times with the exception of two peaks at around 21 and 45 hours, where the NN and model forecast temperatures have comparable variance. Also, the reduction in variance is more substantial at longer forecast times than shorter times. As for bias, the model forecasts suffer from a periodic feature with both positive and negative bias values. The NN removes this bias altogether to a near-zero level; the fluctuations around the bias=0 line of the curve labeled bias_NN are statistical noise. In short, the NN reduces both the bias and variance of the forecasts at (nearly) all forecast hours.

It is possible to view the functional dependence between model and NN forecasts. The NN has 4 inputs: forecast time, model forecast, and two principal components. As such, in order to view the relation between model forecast temperature and the NN output, one must specify the values of the remaining inputs. Figure 6 shows that relationship when the forecast hour is set to 0, 6, 12, 18, and 24, and the inputs are set to their respective average values. The diagonal line is the curve one would obtain if the NN and model forecast temperatures were identical. It can be seen that at 0hr, the NN forecasts differ most from the model forecasts for lower temperature values, and converge for higher temperature values. This suggests that even during the analysis phase the model can benefit from some sort of statistical processing. The NN forecasts at 6, 12, 18, and 24hr, are generally higher (lower) than model forecast for lower (higher) temperature values. Also, note that the nonlinearity of the curves is most notable at longer forecast hours. (The curves for longer than 24hr forecasts are not shown).

The behaviors portrayed above are strongly station dependent. For viewing purposes, the above results must be distilled further. To that end, the MSE, bias, and variance of the forecasts are averaged across all 60 forecast hours. Also, instead of displaying model and NN performance measures separately, the percent improvement of the NN over the model forecast temperatures will be displayed (Figure 7). The numerical values on the x-axis correspond to the labels identifying various stations as set forth in the first paragraph of section 2. The percent improvement in terms of MSE varies between 5% and 85% (Fig. 7, top). On the average, the NN improves model forecast temperatures by 40%-50%. In terms of bias (Fig. 7, middle), the percent improvement is in the 20%-90% range, with the average around 75%. The lowest bias improvement is at a station which is accompanied by the highest variance improvement (Fig. 7, bottom). The variance improvement is generally around 15%-20%. It is then evident that the improvement in MSE brought about by the NN (40%-50%) is mostly due to a reduction in bias ($\sim$ 80%) and somewhat due to a reduction in variance ($\sim$ 20%). These percentages are approximate and reflect both the training and validation values; as expected, the latter are somewhat inferior to the former.

Note that for some stations the percent improvement in terms of variance is actually negative (Fig. 7, bottom). This suggests that the model performs better than the NN in terms of variance. However, in terms of bias, the opposite is true. This yields a near-zero improvement in terms of MSE. In short, for some stations, the model appears to perform adequately, and does not benefit from any post-processing.

6 Summary

Thirty one neural networks are developed for 31 stations. The neural nets successfully improve temperature forecasts produced by The Advanced Regional Prediction System at each and every one of the stations. The improvement is assessed in terms of numerous measures of performance. Although the improvement is evident in terms of both the bias and the variance of the forecasts, the former is more pronounced. The performance across different stations is quite diverse; measured in terms of percent improvement over the model, it varies between 5% and 90% in terms of mean-squared error. Improvement in terms of bias and variance values is in the range 20%-95% and -5%-80%, respectively.

7 Acknowledgement

The author is grateful to Richard Carpenter, Gene Bassett, and Nick Mirsky for providing the data. Richard Carpenter and Dave Montroy are further acknowledged for their review of an early version of the article. Williams Energy Marketing and Trade Company is acknowledged for providing financial support.

8 References

Bishop, C. M., 1996: *Neural Networks for Pattern Recognition.* Clarendon Press, Oxford, pp. 482.

Casaioli, M., R. Mantovani, F. P. Scorzoni, S. Puca, E. Ratini, A. Speranza, 2001: Post-processing of numerical surface temperature with neural networks. Nonlinear Processes in Geophysics; in press. Available at krishna.phys.uniroma1,it .

Glahn, H. R., and D. A. Lowry, 1972: The use of Model Output Statistics (MOS) in objective weather forecasting. *J. Appl. Meteor.*, **11**, 1203-1211.

Efron, B., and R. J. Tibshirani 1993: *An Introduction to the Bootstrap.* Chapman & Hall,436 pp.

K. Hornik, M. Stinchcombe, and H. White, 1989: Multilayer feedforward networks are universal approximators. *Neural Networks*, **2**, 359.

Hsieh, William W., and B. Tang, 1998: Applying Neural Network Models to Prediction and Data Analysis in Meteorology and Oceanography. *Bull. Amer. Meteo. Soc.*, **Vol. 79, No. 9**, pp. 1855.

MacKay, D. J. C., 1996: Bayesian methods for back-propagation networks, in *Models of Neural Networks III*, E. Domany, J. L. van Hemmen, K. Schulten (Eds.), Springer-Verlag, New York, physics of neural network series, pp. 309.

Marzban, C. 2002: Neural Network Short Course, Amer. Meteor. Soc., 2002. Available at http://www.nhn.ou.edu/~marzban .

Marzban, C. 2000: A neural network for tornado diagnosis. Neural Computing and Applications, Vol. 9 (2), 133-141.

Marzban, C., 1998: Bayesian inference in neural networks. Proceedings of the 78th Annual Meeting of the American Meteorological Society, Phoenix, Arizona.

Marzban, C., and A. Witt, 2000: A Bayesian Neural Network for Hail Size Prediction. Accepted for publication by Wea. Forecasting.

Marzban, C., E. D. Mitchell, G. Stumpf, 1999: The notion of "best predictors:" An application to tornado prediction. *Weather and Forecasting*, **14**, 1007-1016.

Masters, T., 1993: *Practical neural network recipes in C++.* Academic Press, 493 pp.

Sarle, W.S., 1994: Neural Networks and Statistical Models. *Preprint in the Proceedings of the Nineteenth Annual SAS Users Group International Conference, Cary, NC: SAS Institute,* pp. 1538-1550. Available at ftp://ftp.sas.com/pub/neural/neural1.ps

Sarle, W.S., 1995: Stopped Training and Other Remedies for Overfitting. *Preprint in the Proceedings of the 27th Symposium on the Interface of Computing Science and Statistics*, pp. 352i-360. Available at ftp://ftp.sas.com/pub/neural/inter95.ps.Z.

Wilks, D. S., 1995: *Statistical Methods in the Atmospheric Sciences.* Academic Press, NY. 467 pp.

Xue, M., K. K. Droegemeier, and V. Wong, 2000: The Advanced Regional Prediction System (ARPS) - A multiscale nonhydrostatic atmospheric simulation and prediction tool. Part I: Model dynamics and verification. *Meteor. Atmos. Physics.*, **75**, 161-193. Available at: http://www.caps.ou.edu/ARPS/ .

Xue, M., K. K. Droegemeier, V. Wong, A. Shapiro, K. Brewster, F. Carr, D. Weber, Y. Liu, and D.-H. Wang, 2001: The Advanced Regional Prediction System (ARPS) - A multiscale nonhydrostatic atmospheric simulation and prediction tool. Part II: Model physics and applications. *Meteor. Atmos. Physics.*, **76**, 134-165. Available at: http://www.caps.ou.edu/ARPS/ .

Table Captions

Table 1. The correlation matrix. The numerical labels refer to the variables as outlined in the second paragraph of section 2.

Figure Captions

Figure 1. The bias of model forecast temperatures as a function of forecast hour at KSAC. The error-bars are standard errors, and the horizontal line is the bias=0 line.

Figure 2. The linear correlation coefficient r between each of the 9 variables described in section 2 and observed surface temperature. The error-bars are standard errors.

Figure 3. Scatterplot of observation vs. NN forecast (a) and model forecast (b) temperatures for a training set (top) and a validation set (bottom).

Figure 4. Residual plots for NN forecasts (a) and model forecast (b) temperatures for a training set (top) and a validation set (bottom).

Figure 5. Bias and variance of model and NN forecasts as a function of forecast time. The top (bottom) curve is for one training (validation) set. Bias is expressed in F° and variance in $(F^\circ)^2$.

Figure 6. The function relating model forecast temperature to NN forecast as represented by the NN. The various curves correspond to different forecast times (from 0hr to 24hr). The diagonal line represents the curve one would get if the model and NN forecasts were identical.

Figure 7. Percent improvement brought about by the NN over model forecast temperatures in terms of MSE (top), bias (middle), and variance (bottom). The solid curve is for a training set, and the dashed curve is for the validation set. The numerical labels on the x-axis correspond to the various stations.

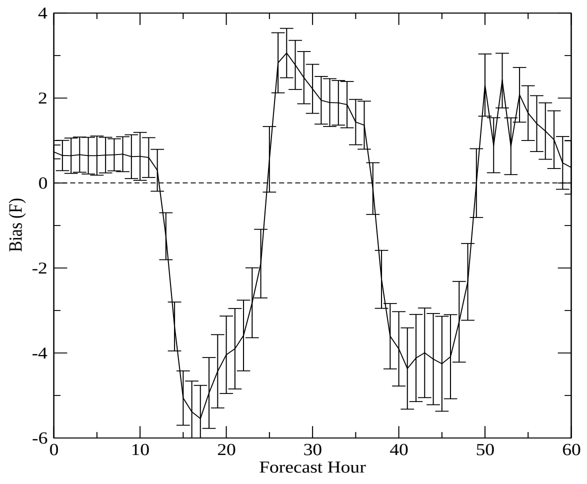


Figure 1.

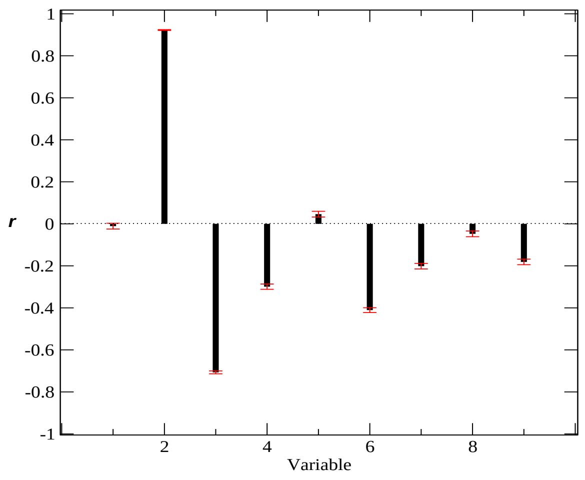


Figure 2.

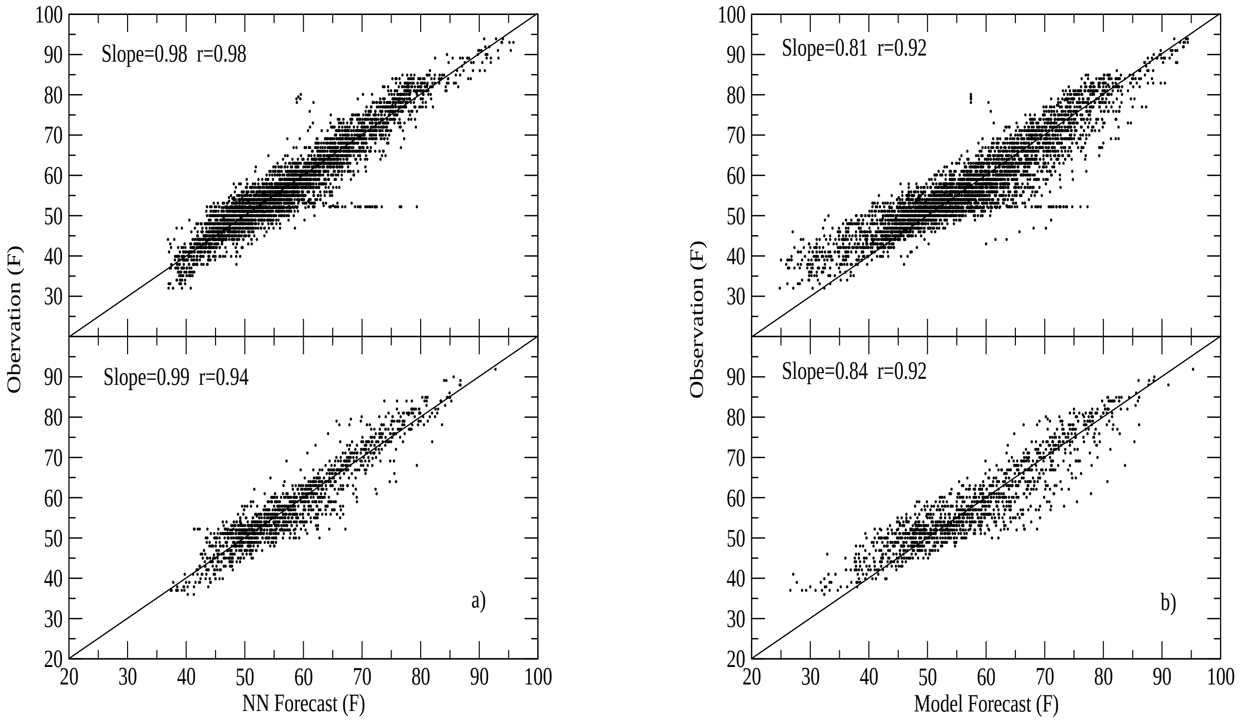

Slope=0.98 r=0.98
Slope=0.99 r=0.94
Observation (F)
NN Forecast (F)
a)
Slope=0.81 r=0.92
Slope=0.84 r=0.92
Observation (F)
Model Forecast (F)
b)

Figure 3.

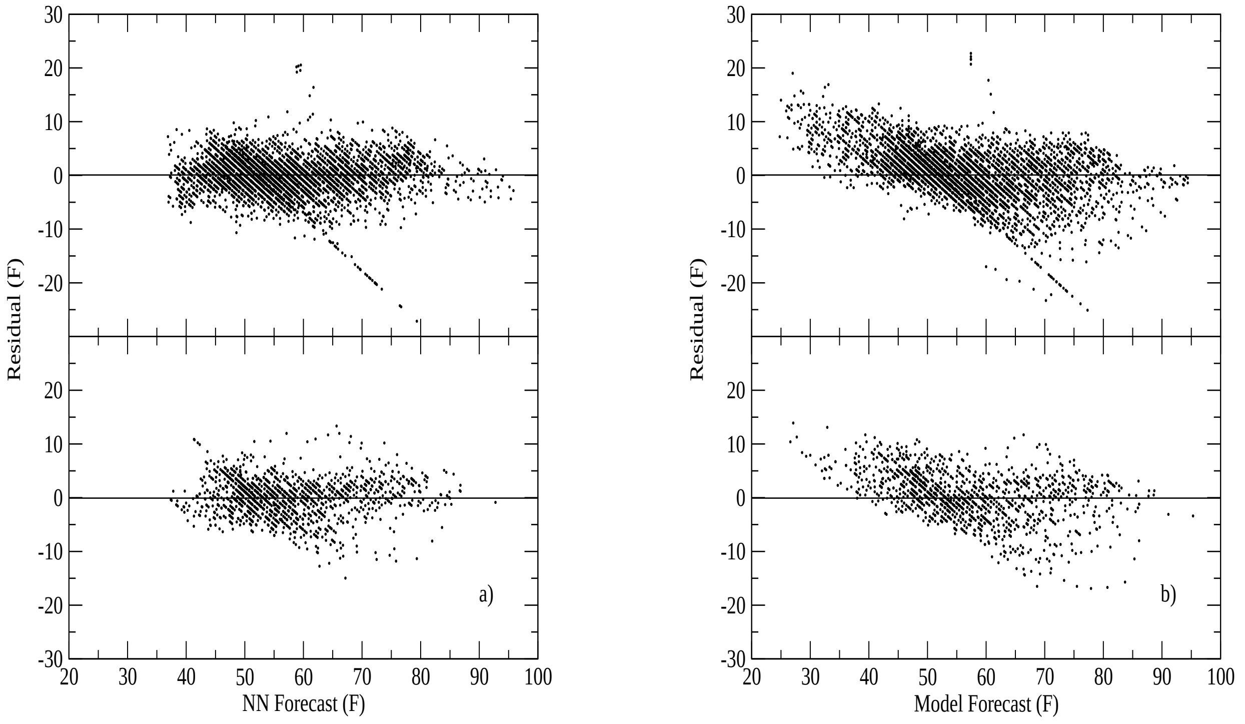
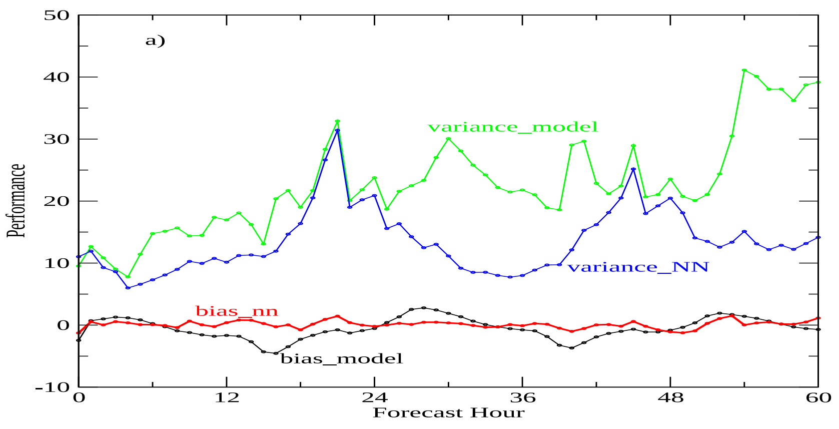


Figure 4.


50
40
30
20
10
0
-10
a)
variance_model
variance_NN
bias_nn
bias_model
Performance
0
12
24
36
48
60
Forecast Hour

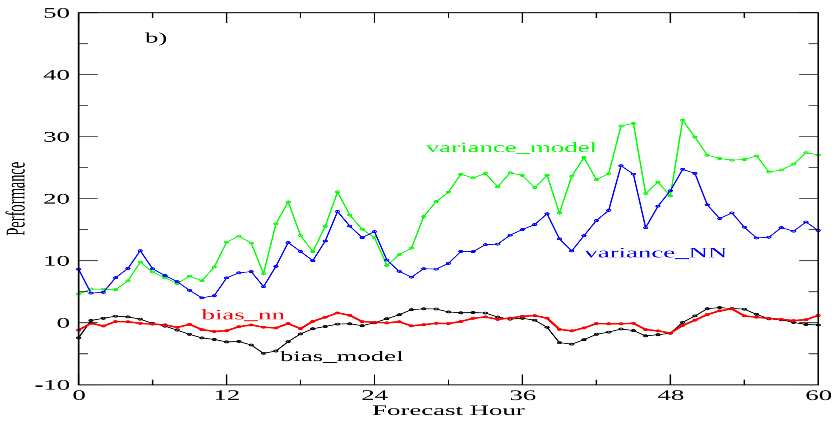
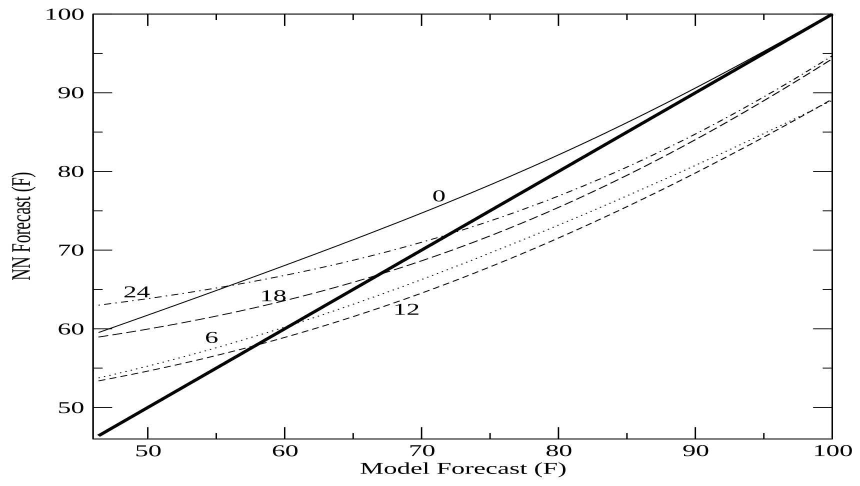

50
40
30
20
10
0
-10
b)
variance_model
variance_NN
bias_nn
bias_model
Performance
0
12
24
36
48
60
Forecast Hour

Figure 5.



NN Forecast (F)
Model Forecast (F)
100
90
80
70
60
50
50
60
70
80
90
100
0
24
18
6
12

Figure 6.

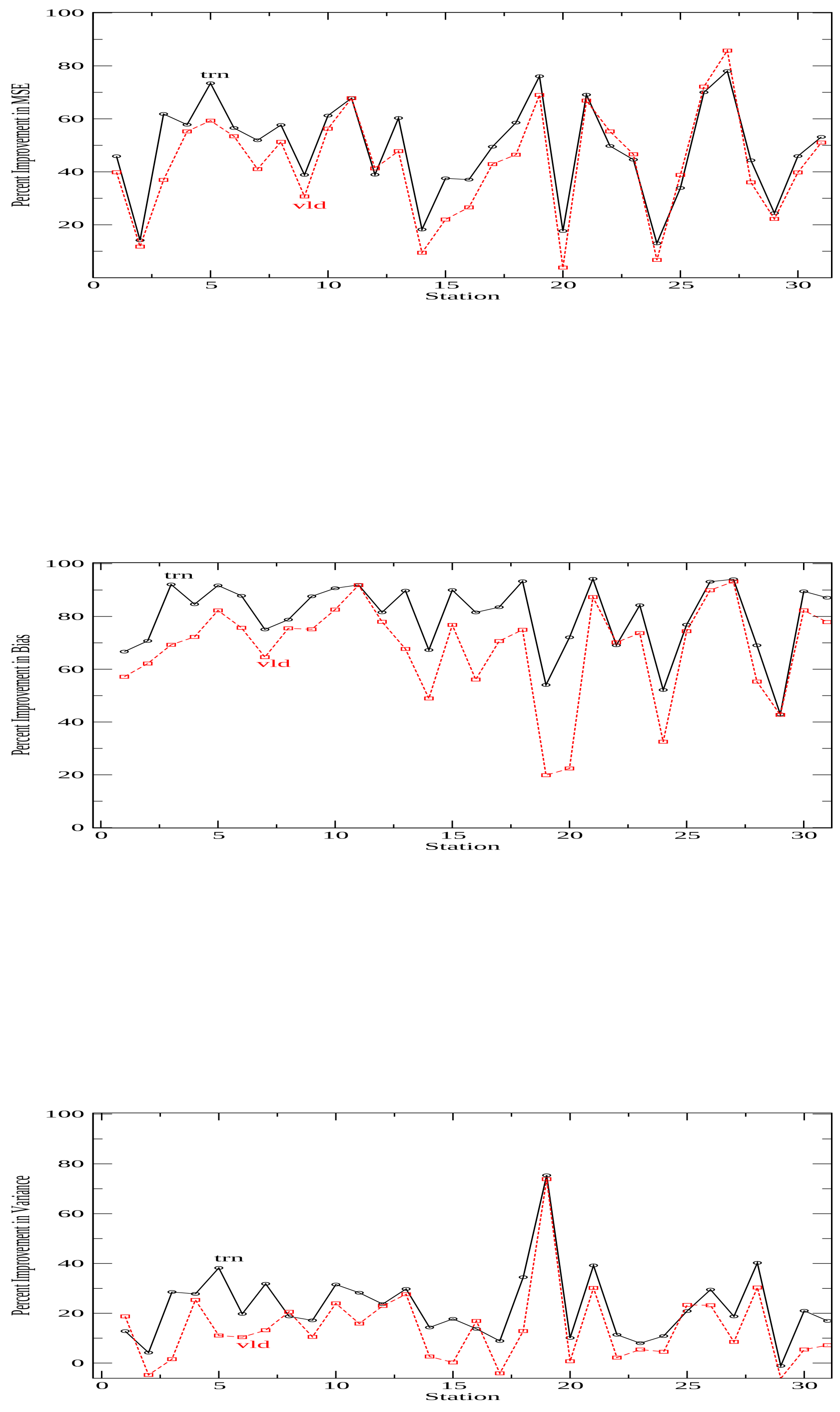

Percent Improvement in MSE
trn
vld
Station
Percent Improvement in Bias
trn
vld
Station
Percent Improvement in Variance
trn
vld
Station
0
5
10
15
20
25
30
20
40
60
80
100

Figure 7.

	1	2	3	4	5	6	7	8	9
1	1.00								
2	-0.01	1.00							
3	-0.03	-0.72	1.00						
4	0.07	-0.35	0.29	1.00					
5	0.04	0.02	-0.24	-0.11	1.00				
6	0.06	-0.52	0.19	0.29	0.13	1.00			
7	0.11	-0.21	0.39	0.03	-0.42	0.04	1.00		
8	0.06	-0.06	0.14	0.12	-0.17	0.01	0.11	1.00	
9	0.20	-0.21	0.30	0.05	-0.15	0.11	0.22	0.30	1.00