NEMEFO: NEural MEteorological FOrecast

Eros Pasero, Walter Moniaci, Tassilo Meindl, Alfonso Montuori
Department of Electronics, Polytechnic of Turin
C.so Duca Degli Abruzzi 24, 10129 Turin (TO), Italy
Phone: +39-011-564-4043, Fax: +39-011-564-4099, Email: eros.pasero@polito.it

Abstract – Artificial Neural Systems are a well-known technique used to classify and recognize objects. Introducing the time dimension they can be used to forecast numerical series. NEMEFO is a "nowcasting" tool, which uses both statistical and neural systems to forecast meteorological data in a restricted area close to a meteorological weather station in a short time range (3 hours). Ice, fog, rain are typical events which can be anticipated by NEMEFO.

I. INTRODUCTION

Typical meteorological forecast systems base their methods on complex mathematical studies rather than large databases coming from satellites and ground weather stations. The approach of NEMEFO is different: it's a " nowcasting " system. A weather station collects meteorological data every fifteen minutes and stores them in a database. The system analyses these data and foresees their evolution up to three hours in advance. NEMEFO can be configured to forecast several parameters such as temperature, humidity, pressure, solar radiation, wind speed and direction and rain. It uses a weather station which samples these data at a given sampling rate. A statistical program, based on the Parzen method [3], evaluates the cross correlation among these parameters and selects the best groups, in the historical database, suitable to forecast the next three hours. This method is shown in section III. These data are used to train an artificial neural network [2], shown in section IV, able to forecast numerical series. The time is added to this network to allow us to evaluate a temporal evolution. The system has been tested on a five years database of actual data coming from an agricultural weather station. A set of data was not used to train the network but was used to test its performance. Results are shown in section V.

II. A SHORT HISTORY

The project begun with the goal to predict the ice formation on a mountain highway. The idea was to use a " data driven" approach to allow security systems to prevent the ice formation in advance. Unfortunately data driven systems need the "ice event" to estimate its occurrence and nowadays this information isn't available because no ice sensor is available on the market. Therefore we decided to use an analytical model to verify the condition of ice on the road. This model is based on six parameters:

Air temperature
Relative humidity
Wind velocity
Solar radiation
Road temperature
Precipitation

We decided to evaluate the evolution of these data to forecast the ice formation three hours in advance.

III. STATISTICAL ESTIMATOR

As we said in the introduction, the statistical tool's [7] goal is to choose the better predictors to predict the meteorological data. The concept used is that of entropy; the formula is the following:

$$e(x) = -\int_{-\infty}^{+\infty} \log(p(x)) \cdot p(x) dx \quad (1)$$

In this formula $p(x)$ is the Probability Density Function (PDF) of a random variable x. $e(x)$ is an index of dispersion that lies in the range $]-\infty,+\infty[$. Let Z be the vector of all the possible predictors that can be used to foresee the variable y (predictand). Now let X_1 and X_2 be two particular subsets of predictors taken from Z. The number of elements in X_1 and X_2 has to be the same. In order to establish the best set of predictors between X_1 and X_2 we calculated the following entropy difference:

$$\begin{aligned} d(X,y) &= e(X,y) - e(X) = \\ &= -\int \log P(X,y) \cdot P(X,y) dX dy + \\ &+ \int \log P(X) \cdot P(X) dX \end{aligned} \quad (2)$$

where p(X,y) is the joint PDF of X and y and p(X) is the PDF of X (the predictors). But the problem is that both PDFs are unknown, so that it's impossible to evaluate the integrals in (2). We circumvented this problem by estimating the unknown PDFs through the Parzen [3] method. This method estimates the unknown probability density functions making a sum of Gauss kernels, each one centred on a record of the historical database. The formula is the following:

$$P_X^*(X;\mathbf{D},\Lambda) = \frac{1}{n}\sum_{i=1}^{n} \prod_{j=1}^{m} \frac{1}{\sqrt{2\pi\lambda_j^2}} \cdot \exp\left[-\frac{(x_j - x_{ij})^2}{2\lambda_j^2}\right] \quad (3)$$

where,

- **D**={$\mathbf{X_1}$,....,$\mathbf{X_n}$}: it's the observations' vector
- n: database dimension (number of records)
- x_i,x_{ij}: these are the j-th component of X and X_i
- Λ: standard deviation vector, $\Lambda = (\lambda_1, \ldots, \lambda_m)$

Each standard deviation in Λ regulates the resolution of the estimator along the corresponding dimension.

In its turn this allows us to estimate – say – d(X_1,y) and d(X_2,y). The best between X_1 and X_2 is the one that gives rise to the smallest entropy difference.

For example for the prediction of the relative humidity one hour ahead, we found that the best predictors for the network were:

Humidity value one hour before the forecast instant
Humidity value two hour before the forecast instant
Humidity gradient of the last four hours
Air temperature gradient of the last four hours
Air temperature value two hour before the forecast instant

IV. ARTIFICIAL NEURAL NETWORK

A neural network [1], considering the timing evolution of these parameters, seems to be the best approach to foresee time series data and the probability of ice formation.
The system profile can be modelled as the output of some dynamic system, influenced by weather variables, time and other environmental variables. A neural network with feedback can simulate a discreet time dynamically system. The general feed-forward topologies with weights sharing can represent feedback connections by unfolding in time the basic network. Of course using this technique, feedback can be followed for a finite time (also arbitrary large). In this way general feed-forward networks can simulate the dynamic systems, but only for their transient behaviour. It's necessary to define a network topology (*equations and connections*). The relation between the network input and output depends on several weights that can be modified. Then we must fix a *training rule* in order to adjust the weights and, consequently, to reduce the difference between the network output and the real value (*target*). We have used a feed-forward network with three layers:

Input layer
Hidden layer (5 neurons)
Output layer (1 neuron)

Each neuron has, as transfer function, the hyperbolic tangent function

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \qquad (4)$$

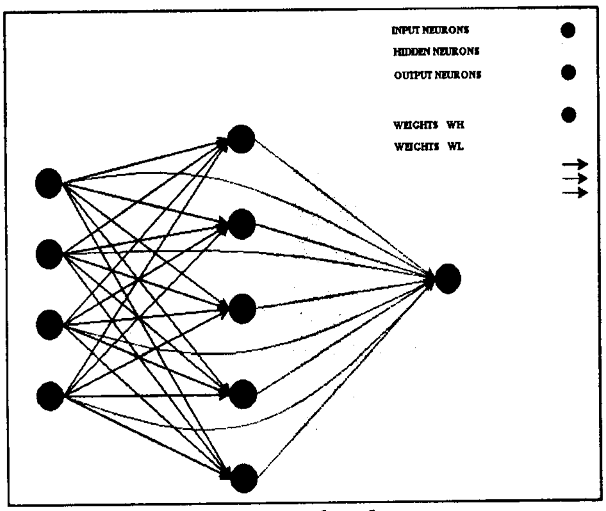

Figure 1: Neural network scheme

We decided to use backpropagation rule to train the network. In this method we choose the weights in order to minimize the square error on training set

$$E = \sum_{t=1}^{T} E(t) = \sum_{t=1}^{T} \sum_{i=1}^{n} 0.5 \cdot [\hat{y}_i(t) - y_i(t)]^2 \qquad (5)$$

The singularity of backpropagation [2] is how this expression is minimized. First of all we fix the weights of the network. It's usual to choose random numbers between –0.1 and 0.1. Then we calculate the network output $\hat{y}(t)$ and the errors E(t) with the previous weights. After we consider the derivative of E as to the weights:

$$F\_W_{ij} = \sum_{t=1}^{T} F\_net_i(t) \cdot x_j(t) \qquad (6)$$

In this way we adjust the network weights, moving towards the opposite direction to the gradient. The formula used is the following:

$$W_{ij} = W_{ij} - learning\_rate \cdot F\_W_{ij} \quad (7)$$

The first step has been to simulate a real meteorological station using real data both for training and test phase. We used the database from the meteorological station of Spilimbergo (PN), Italy. The parameters observed are the following:

Time
Air temperature (2 meters of altitude)
Relative Humidity
Wind velocity
Terrain temperature (10 centimetres of depth)
Solar radiation
Precipitation

The sampling is hourly. We have collected the data during these periods:

January – February (1998)
November – December (1998)
January – February (1999)
November – December (1999)
January – February (2000)
November – December (2000)
January – February (2001)
November – December (2001)
January – February (2002)

We used 10056 training patterns, hourly sampled from January (1998) to February (2001) whereas other 2880 patterns sampled from November (2001) to February (2002) were used for test.

V. RESULTS

In order to judge the forecast quality, we calculated the VRC (Variance Reduction Coefficient) in this way:

$$VRC = \frac{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}{\frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2} \qquad (8)$$

Numerator represents the mean square error whereas denominator stands for the variance of the training set. VRC is very meaningful as it puts in evidence how much meteorological parameter variability is explained by network inputs used. The more VRC is small the more the prevision is careful. In the following table we reported VRC values for each prevision and each meteorological parameter.

Table I : VRC values for each prediction

	VARIANCE	VRC 1 hour	VRC 2 hours	VRC 3 hours
Air Temperature	27,99 [(°C)^2]	3,6%	8,7%	13,9%
Relative Humidity	479,88	4,7%	13,2%	20,9%
Terrain Temperature	11,37 [(°C)^2]	0,026%	0,063%	0,082%
Wind Velocity	1,39 [(m/s)^2]	34%	48,7%	60%
Solar Radiation	11200 [(W/m^2)^2]	5,6%	13,2%	19,8%
Precipitation	0,33 [(mm)^2]	7,2%	14,2%	19%

The results obtained are good; only wind velocity case isn't satisfactory. This bad result is due to the small temporal scale of wind velocity in comparison with the forecast interval. As example, we report the graphics relative to air temperature, humidity and rain forecasts.

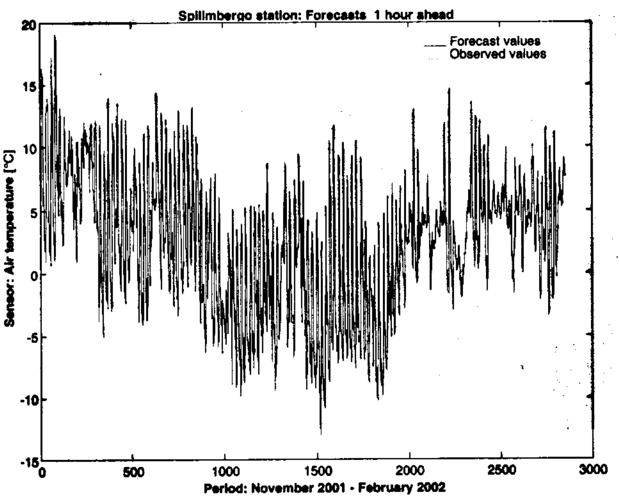

Figure1: Air temperature forecasts 1 hour ahead

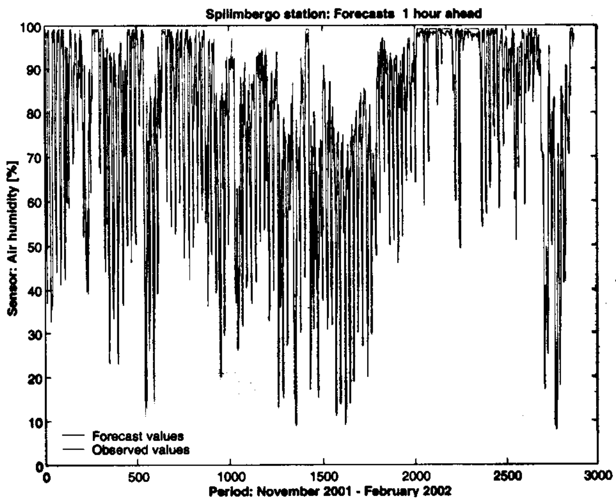

Figure2: Air humidity forecasts 1 hour ahead

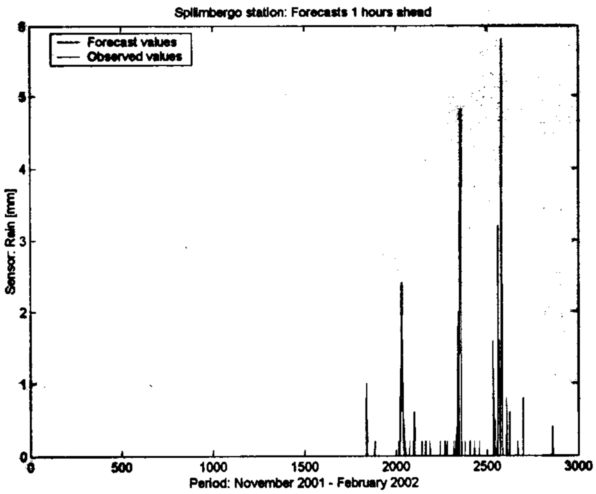

Figure3: Rain forecasts 1 hour ahead

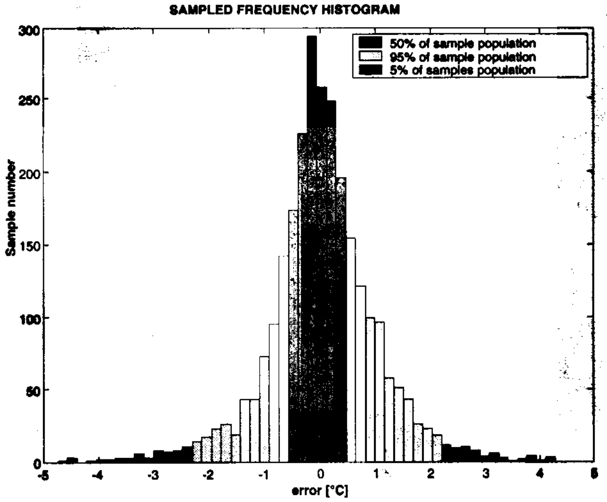

Figure4: sample frequency histogram (temperature)

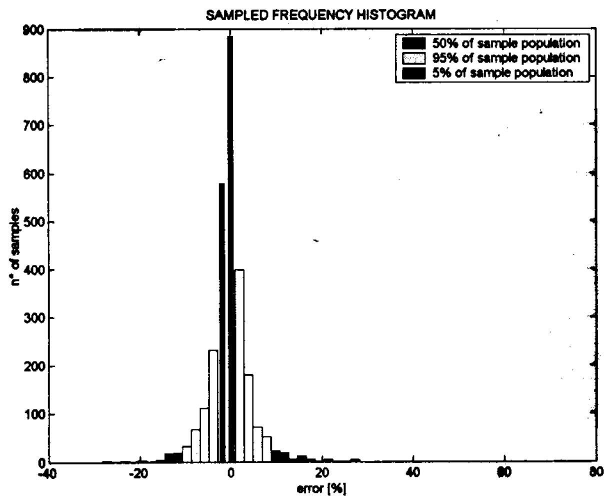

Figure5: sample frequency histogram (humidity)

From the previous graphs we can see that the forecast's errors are very little, and in particular:

- *Air temperature: 3.6%*
- *Air humidity: 4.7%*
- *Rain: 7.2%*

Figure 4 shows that 50% of forecast population (*air temperature*) has an error included between ± 0,5°C whereas 95% between ± 2°C. Regards to humidity the 50% of forecasts has an error between ± 4.5% and 95% between ± 10%. Knowing that the accuracy of the humidity and temperature sensors are respectively ±3.5% and ±0.5%, these results are very good. We can improve performances, increasing the training patterns number. As regard as the evaluation of ice formation probability we used an analytical model [7]. This model calculates the amount of ice that is present on the road at the end of the forecast period. It needs receiving the forecast values of the meteorological parameters by the forecast neural system described in the previous paragraph. Four modules make it up:

1st module: it [4,5,6] calculates water plus ice mass balance on the road.

2nd module: it [4,5,6] reckons water minus ice mass balance on the road. This modulus is used only when the road temperature is 0°C. In fact only in this case it's possible the water change phase. As far as this process is concerned it's very important the energy balance on the road.

3rd module: it calculates the snow heap and the snow melting on the road.

4th module: this module, according to the air – asphalt interface temperature, integrates opportunely the previous modules. On the starting integration instant we used the observed meteorological parameters whereas on the last instant we used that one forecast.

As we hadn't the interface temperature in the database, we calculated it with the following formula,

$$T(z) = T_0 + \left[\frac{T_{210} - T_0}{210 \cdot Rapp}\right] \cdot z \qquad (9)$$

where,

T_0 : road temperature (10 centimetres of depth)
T_{210} : air temperature (2 meter of altitude)

Putting z=10 in the formula, we get the interface temperature. The coefficient *Rapp* represents the ratio between asphalt and air thermal volumetric capacity. *Rapp* is necessary because we aren't in a homogeneous system. We tested the model on the period November (2001) – February (2002).

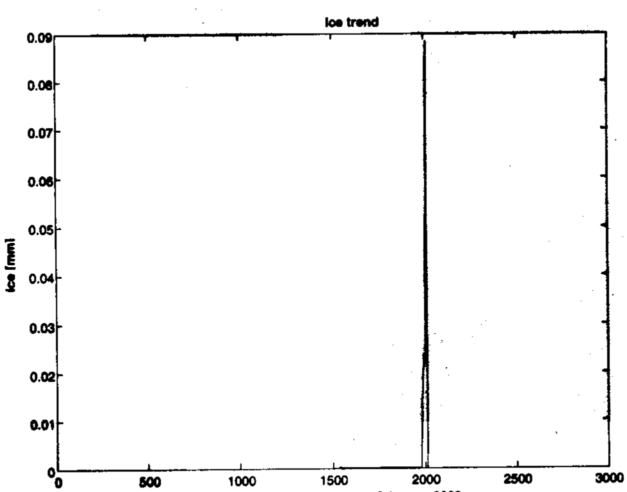

Figure 6: Ice trend

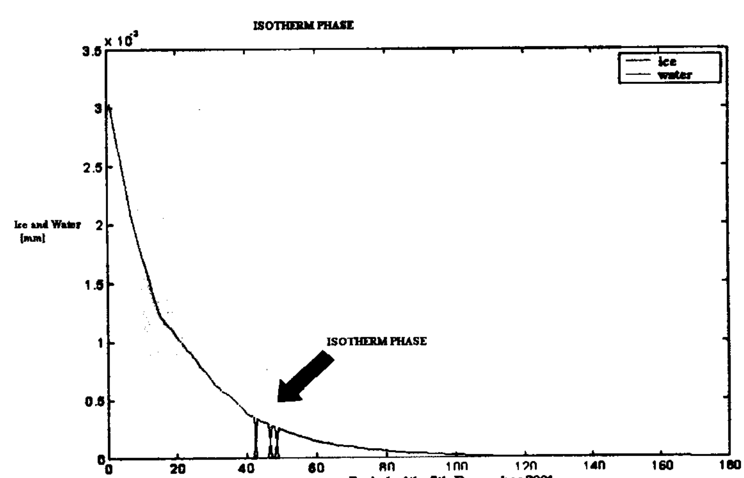

Figure 7: Isotherm phase (water and ice)

VI. CONCLUSION

Nowcasting seems to be a new interesting topic in meteorological forecast. The possibility to have an accurate estimate of the weather conditions in next three hours can be a useful tool for several situations, such as ice on the road, fog, rain and so on. A powerful software, NEMEFO, was shown in this paper. It's now used in some prototype weather situations in Italy and gives a good performance. A reduced version is visible 24 hours a day at www.neuronica.polito.it. Next version of NEMEFO will integrate the forecast of a network of stations to evaluate weather forecast for a large area.

ACKNOWLEDGEMENTS

This work was partly funded by MIUR research project n.2002095532_002_. Thanks to Mario Costa, the original inventor of NEMEFO, who isn't more with us.

REFERENCES

1.W. Moniaci, "Studio e Sviluppo di un Sistema Neurale Artificiale per la Previsione della Formazione del Ghiaccio sulle Strade". Graduation thesis, Polytechnic of Turin – I, 2002.

2. P. Werbos, "Beyond regression: New Tools for Prediction and Analysis in the Behavioural Sciences". Ph.D. dissertation, Committee on *Appl.Math.*, Harvard Univ., Cambridge, MA, Nov. 1974.

3. E. Parzen, "On Estimation of a Probability Density Function and Mode". *Ann. Math. Stat.*, Vol.33 , pp. 1065 -1076, 1962.

4. B. H. Saas, "A Numerical Model for Prediction of Road Temperature and Ice". *J Appl. Meteor.*, Vol.36 , 801-817, 1992.

5. P. Crevier and Y. Delage, "METRo: A New Model for Road – Condition Forecasting in Canada". *J Appl. Meteor.*, Vol. 40 , 2001.

6. Y. Delage, "Parameterizing Sub-grid Scale Vertical Transport in Atmospheric Models under Statically Stable Conditions". *Bound.-Layer meteor.*, Vol. 82 , 23-48, 1997.

7. Mario Costa, Walter Moniaci, Eros Pasero, "INFO: An Artificial Neural System to Forecast Ice Formation on the Road", International Symposium on Computational Intelligence for Measurement Systems and Applications (CIMSA) Lugano, Switzerland, 29-31 July 2003.