

Coupling fuzzy modelling and neural networks for river flood prediction

Giorgio Corani*, Giorgio Guariso

Dipartimento di Elettronica ed Informazione, Politecnico di Milano.

e-mail: {corani, guariso}@elet.polimi.it

This is a preprint of the paper published on
IEEE Transactions on on Men, Systems and Cybernetics part C,
35(3), 382-391, 2005

Available from <http://ieeexplore.ieee.org/>

*Corresponding author

Abstract

Over the last decade, neural networks-based flood forecasts systems have been increasingly used in the hydrological research. Usually, input data of the network are composed by past measurements of flows and rainfalls, without providing a description of the saturation state of the basin, which in contrast plays a key role in the rainfall-runoff process. This paper couples neural networks and fuzzy logic in order to enrich the description of the basin saturation state for flood forecasting purposes. The proposed framework first classifies the basin saturation state through fuzzy logic, and then issues the forecast exploiting a set of neural predictor, each trained for a certain basin saturation condition. The outputs of the specialized neural predictors are linearly weighted, according to the basin state at forecast time: the more the training conditions of a predictor matches the current basin saturation state, the higher its weight on the final forecast. The basin state is assessed analyzing the total rainfall occurred on a certain time window before the flood event. The framework has been tested on an Italian catchment and may overperform classical neural networks approaches.

keywords: feedforward neural networks, fuzzy sets, forecasting, rivers, hydrology

1 Introduction

An efficient flood alarm system may significantly improve public safety, and mitigate economical damages caused by inundations. Flood forecasting is undoubtedly a challenging field of operational hydrology, and a huge literature has been developed in years; in particular, the rainfall-runoff relationship has been recognized to be non linear.

Over the last decade, artificial neural networks (ANN) have been increasingly used in the hydrological forecasting research (see for instance [14], where tens of papers on the topic are quoted). They do not substitute conceptual watershed modelling, since they cannot represent the internal structure of the watershed, neither can easily manage distributed data regarding the physical properties of the basin. However, they have been recognized as a viable alternative to conceptual models for input-output simulation and forecasting [19], and usually allow to shorten the time spent in developing the model [22]. Furthermore, their computational speed in simulating and forecasting is very welcomed in real time operations.

During high flows episodes, the spatial and temporal distribution of the rainfall field rules the flood waves formation, while other hydrological processes (such as evaporation and transpiration from water bodies and surface, or aquifer contribution to the flow) become of little relevance. Therefore, the accuracy of the hydrological forecast depends heavily on the accuracy of the rainfall field measurement and representation [12]. It should be remarked that only some precipitation (*effective rainfall*) joins directly the river, while the remaining part is infiltrated and stored by the soil, to eventually reach the river at much later times, exceeding the flood duration. During heavy rainfall events, the basin saturates progressively, and this causes the infiltration process to become less effective, routing a higher fraction of effective rainfall to the river. A well known issue is that the catchment response to rainfall impulses may dramatically change depending on the saturation degree of the basin [20], which constitutes a critical modelling question. In fact, the estimate of infiltration parameters (and consequently of the effective rainfall) is recognized as a major uncertainty source in physically based models [15]. Piecewise approaches [21, 23, 10] have been proposed to represent the rainfall-runoff relationship switching among different models, depending on an indirect measure of the soil saturation state, usually obtained from antecedent precipitations.

Neural network inputs usually represent the autocorrelation of the time series and the precipitation field through past measures of flows and rainfall. Although autoregressive terms could act as proxies for the saturation state (the higher the past flows, the more the basin is expected to be saturated), such representation may become inadequate in heavy rain cases [4]. It should also be remarked that the basin saturation is not directly observable, since it would require a huge number of real-time "in situ" samplings. In order to account for the saturation state, we adopt a proxy constituted by the total rainfall measured on the basin in the days preceding the prediction time and, since the use of a proxy rather than an actual measure could cause some degree of noise in the information, we developed a fuzzy approach for basin state assessment. At each time instant, the proxy based on the preceding rainfall is computed and then the basin state is classified in a fuzzy manner, through a set of membership degrees with respect to different saturation classes.

Assuming that each saturation class results in a different non linear rainfall-runoff rela-

tionship, we build a set of neural networks, each targeted to a specific saturation condition. The specialized networks are trained according to a weighted least square criterion, i.e. the training objective function of the j -th network weights each square error by the membership of the basin with respect to the j -th saturation class. Data having a null memberships with respect to the j -th class, i.e. representing basin conditions far from those on which we want to specialize the model, are hence disregarded during training. The forecast is finally issued as a weighted sum of the specialized models outputs; weights are actually constituted by the current memberships of the basin state to the saturation classes. Abrupt model switching in correspondence of the thresholds, which are a drawback of piecewise approaches, are avoided since we smoothly change the weight of each model on the final prediction according to the basin state.

Combinations of ANN have been already subject of investigation: Hashem and Schmeiser [9], for example, propose an algorithm which optimizes the coefficient used to linearly combine the output of several neural networks, while Elragal [6] combines through an ANN the forecasts issued by three further ANNs, different in topology, training algorithm and input variables in order to predict the natural gas load.

Combination of estimators have been also used in the hydrological field. Shamseldin et al. combined the output of miscellaneous rainfall-runoff models (black, grey box and lumped conceptual ones) through statistical methods such as weighted mean, neural networks [18] and fuzzy logic [25]. Remarkably, each model was calibrated to globally represent the rainfall-runoff relationship; differently, our framework will issue the prediction by collecting the outputs of specialized models, combining them on the base of the state of the basin.

The paper is organized as follows: we first describe the methodological framework (Section 2 to 5), i.e. the neural network model of the rainfall-runoff relationship, the fuzzy representation of the basin state, and the calibration procedure adopted. Then, we compare the forecast accuracy of the system with a classical neural network approach for a river basin located in the Northern Italy (Section 6).

2 Neural network modelling of the rainfall-runoff relationship

The proposed forecast system is based, according to a typical hydrological approach, on the availability of rain gauges distributed within or near the watershed. The forecast are issued after the arrival of the rainfall events; since rainfall-runoff response times are of the order of few hours for small and medium sized basins (i.e. under or about 1000 km^2), this is a natural bound on the forecast lead times. In a recent work [11], lead times have been successfully increased up to 24 hours even on small basins, acquiring data from radar, radiosondes and satellite, thus monitoring on a wide area the synoptic evolution of the atmospheric conditions. However, the river basin studied in the paper, as many others, is simply monitored by means of hydrometers and rain-gauges, which will indeed provide the input used for computing the predictions.

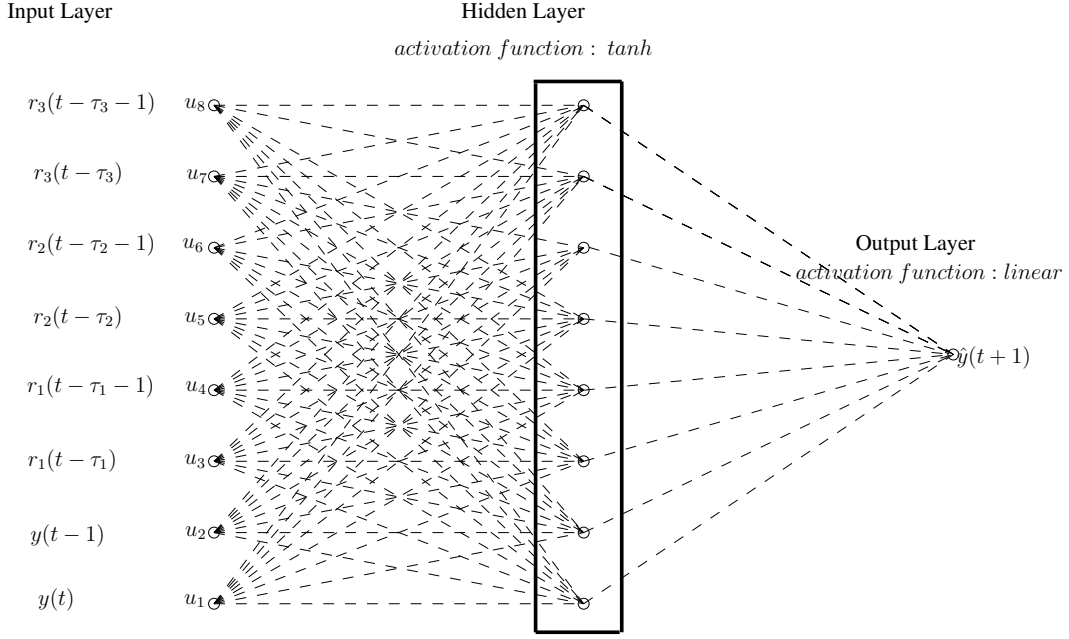


Figure 1: Structure of a fully connected neural network.

We model the rainfall-runoff process through a feed forward neural network with one hidden layer, assuming the water level $y(t+k)$ as target variable to be predicted. The model input set comprises an *autoregressive* part of order a (i.e., a past water level measurement taken at the hydrometer), and several rainfall variables, associated with the m available rain gauges r_1, r_2, \dots, r_m . Rainfall inputs are delayed by time lags $\tau_1, \tau_2, \dots, \tau_m$, in order to take into account the travel times taken by the rainfall to join the river running on the land surface. Hence, assuming an equal order e on each gauge input, the *exogenous* variables are constituted by rainfall measures $[r_1(t-\tau_1), r_1(t-\tau_1-1), \dots, r_1(t-\tau_1-e+1), \dots, r_m(t-\tau_m), \dots, r_m(t-\tau_m-e+1)]$; therefore, the network input set collects overall $(a+m \cdot e)$ variables. We define u as the vector containing all the input variables. Figure 1 shows a sample neural network structure with 3 rainfall gauges, 8 nodes in the hidden layer, having order $a = e = 2$ on both autoregressive and exogenous variables.

Within the network, the j -th node of the hidden layer sums the inputs data as follows:

$$z_j = \sum_{i=1}^{i=a+m \cdot e} w_{ij} u_i - b_j \quad (1)$$

where w_{ij} is the weight of input u_i at the j -th node, and b_j is the neuron bias. The signal z_j becomes then argument of the neuron activation function (namely, hyperbolic tangent):

$$c_j = f(z_j) = 1 - \frac{2}{(\exp(2z_j) + 1)} \quad (2)$$

The outputs c_j of neurons in the hidden layer are then sent to the output layer; it contains a unique linear node which weights the values c_j by W_j and returns the forecast:

$$\hat{y}(t+k) = \sum_j W_j c_j - b_{out} \quad (3)$$

where b_{out} represents the bias associated to the output neuron. The network is configured as a direct predictor, i.e. it returns the k -steps ahead prediction avoiding the intermediate forecast steps required by recursive models, which may result in errors propagation. In fact, under mild assumptions, direct predictors can be demonstrated to provide higher performances than recursive ones [1].

While notation in formula (3) is proper when dealing with a unique ANN model, we have to apply some changes to manage at the same time the output of several networks. Hence in the following we will denote by $\hat{y}_j(t+k)$ the forecast issued by the j -th specialized model, and by $\hat{y}(t+k)$ their fuzzy combination, which is the actual prediction of the framework.

3 Basin saturation issues

Let us define the *cumulated rainfall on time window Ψ* -in the following denoted as $R(\Psi, t)$ - as the integral of the rainfall field considered on the whole basin area over the time window $[t - \Psi, t]$. Actually, $R(\Psi, t)$ is an unknown information since only instant measures taken at the individual gauges are available; we use the Thiessen interpolation method ¹ to reconstruct the rainfall field on the whole basin. This information is obtained at no costs elaborating rainfall measures which are already available, and does not involve any significant computational overload; despite these interesting properties, such an information does not appear to be widely used in hydrological applications of ANN. In the following, we adopt $R(\Psi, t)$ as a proxy for basin saturation state; since the use of a proxy involves some uncertainty, we use a fuzzy approach to evaluate the basin state.

First, the C-means fuzzy clustering algorithm [2] is applied on the calibration time series of $R(\Psi, t)$ to identify a set of centroids $[C_1, C_2, \dots, C_n]$. For example, in the case of three centroids, C_1 can be thought as representative of the low cumulated rainfall condition (i.e., “dry” basin), C_2 of an intermediate situation and C_3 of the high cumulated rainfall class (“wet” basin).

Then, we assign standard membership functions (triangular and trapezoidal) to map each point of the cumulated rainfall domain to a set of membership degrees with respect to the different saturation classes. Cumulated rainfall values comprised between two centroids belong with a not null membership degree to both the classes; the membership increases [decreases] linearly as values become closer [farther] to [from] a certain centroid. Classes spanning the beginning or the end of the domain are given trapezoidal membership functions: hence, values lower than the first centroid, or higher than the last one, have an identical membership 1 with respect to the first [last] class.

¹The Thiessen method subdivides the catchment area into several polygons, whose boundaries are located at a distance halfway between gauge pairs. The rainfall is then assumed to be constant over each polygon and $R(\Psi, t)$ is finally estimated as:

$$R(\Psi, t) = \sum_{\tilde{t}=t-\Psi}^{\tilde{t}=t} \sum_{i=1}^{i=m} s_i r_i(\tilde{t}) \quad (4)$$

where s_i is the ratio of the area of the i -th polygon to the whole catchment.

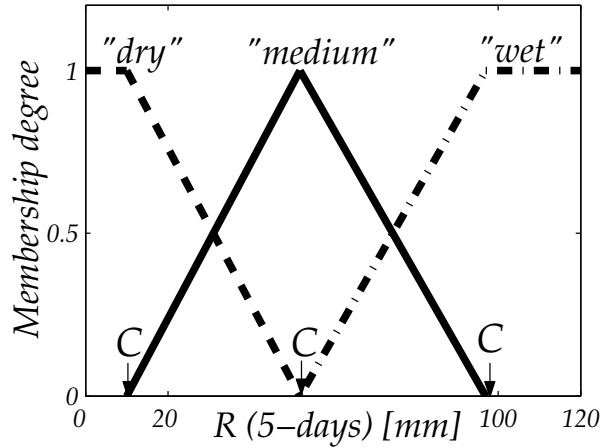


Figure 2: Sample of membership functions with three centroids.

For instance, in the sample presented in Figure 2, a cumulated rainfall of 20 *mm* corresponds to a 0.75 fuzzy membership to the first class and 0.25 to the second class; we can therefore state that it represents a “quite dry” situation. In our framework, each saturation class has an associate neural network model, under the rationale that different saturation conditions result in different non linear rainfall-runoff relationships.

To properly select a time window Ψ which makes cumulated rainfall suitable as proxy for the basin state, we perform an hydrological analysis. If a function mapping water levels and flows is available, one could compute the integrals of observed rainfall and flows on the event; indeed, their ratio, usually referred as rainfall-runoff coefficient, gives an indication about the effectiveness of the infiltration process. In our case a water level-flows function is not available; we thus compute a ratio λ , using the sum of observed water levels as a rough substitute for the flow integral. In principle, the more saturated the basin, the less the rainfall infiltration, the greater the increase on runoff. Indeed, looking at two samples (Figure 3a,b) taken from the Olona river study (presented later in the paper), the ratios λ computed on the different floods show an increasing relationship with the estimate of $R(\Psi, t)$ at the beginning of the events.

In particular, λ shows an almost linear dependence ($\rho = .81$) with respect to the 5-days cumulated rainfall, reaching its maximum in correspondence of the cumulated rainfall maximum. On the other hand, a much more spread relationship is detected ($\rho = .38$) by using the 2-days cumulated rainfall. Thus, the 5-days time window appears to be more suitable in representing the basin saturation state. Cumulated rainfall seems to be a poor information when it has a null value: indeed, in this case the basin can be effectively dry, or affected by rainfall occurred outside the time window Ψ . As a matter of fact, null values of $R(\Psi, t)$ result in spread values of λ . However, the great variability of λ demonstrates how an identical storm may perturbate the observed runoff in a totally different manner, depending on the basin saturation state

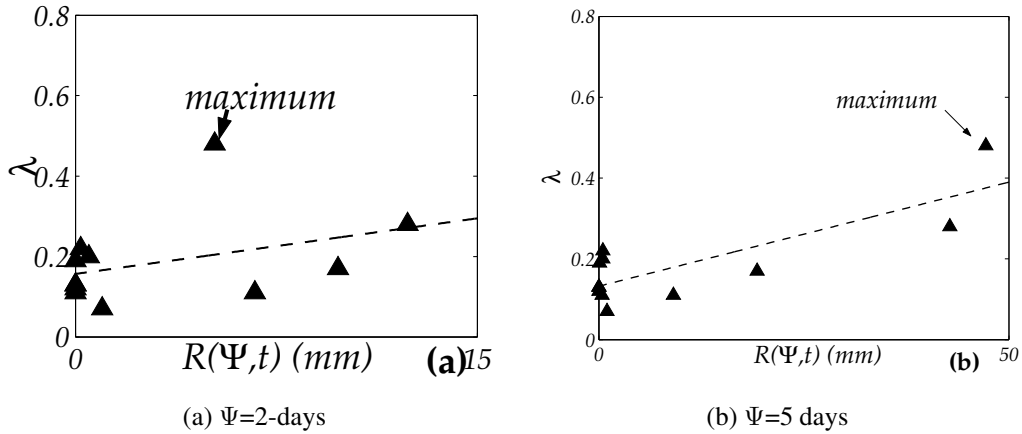


Figure 3: λ as a function of cumulated rainfalls at the beginning of the floods. Dashed lines show the linear interpolations.

4 Framework overview

According to the split sample approach [3], we divide the available data into three different subsets, ensuring as far as possible the similarity of statistical properties between them:

- a training set \mathcal{T} , with cardinality N , used in the parameters identification;
- a validation set \mathcal{V} , with cardinality M , used to implement *early stopping* [3]. The union of \mathcal{T} and \mathcal{V} corresponds to the whole calibration set, in that the two sets are jointly exploited in the predictor identification;
- a testing set \mathcal{S} , used to assess the model performances on previously unused data.

Since data standardization makes the training algorithm numerically robust and leads to a faster convergence [13], mean and variance of the training time series are used to standardize the three data sets.

The whole identification procedure of the framework could be briefly schematized as follows:

1. splitting of the data into training, validation and testing set;
2. data standardization;
3. selection of the time window Ψ , used to compute the time series $R(\Psi, t)$;
4. identification of the cumulated rainfall centroids;
5. definition of the triangular and trapezoidal fuzzy membership functions;

6. identification of a specialized neural network for each saturation class.

It might constitute a challenging issue learning in an integrated way the parameters of the memberships functions and of the specialized local predictors. An interesting attempt in this direction has been carried out by [26] for fuzzy systems with specialized linear predictors. Nevertheless, an efficient algorithm able to jointly calibrate membership functions and non linear regressors, as in the case at hand, is not available at present and would probably require heavy computation times.

5 Specialized neural networks predictors

Neural networks are usually trained according to a least square criterion, and thus the parameter estimate tends to capture the average situation in the dataset. However, we are interested here in specializing a different model to each saturation condition; therefore, we adopt a weighted least square criterion. Let us denote with $\mu_j(t)$ the membership degree of the basin saturation state at time t with respect to saturation class j , and with $\hat{y}_j(t+k)$ the output of the j -th network. The weighted least square function for the j -th predictor can be written as:

$$\Phi(\theta) = \frac{1}{2N}D \|\theta\| + \frac{1}{2N} \left(\sum_{t \in \mathcal{T}} \mu_j(t) [y(t+k) - \hat{y}_j(t+k, \theta)]^2 \right) \quad (5)$$

Hence, if $\mu_j(t) = 0$, the corresponding error does not affect the objective function nor the parameters estimate of the j -th predictor, and the network will not learn hydrological situations too far from its own saturation class. On the other hand, as $\mu_j(t)$ increases, the impact of the error on the objective function becomes higher. The objective function uses the basin classification $\mu_j(t)$ for the k -steps ahead forecast $\hat{y}(t+k)$, since values $\mu_j(t+1) \dots \mu_j(t+k)$ are clearly unknown at time (t) . In formula (??) a standard regularization term, proportional to the norm of the weights $\|\theta\|$ through the weight decay coefficient D , is added since it has been demonstrated to improve the robustness of weights identification [8].

The minimization of the objective function (??) has been accomplished through a variant of the Levenberg Marquardt algorithm, which we realized starting from the standard least square implementation provided in the Neural Network Based System Identification Toolbox for Matlab [16]². The training algorithm we implemented is described in Appendix A.

As already anticipated, we adopted early stopping [3] to provide the termination condition to the training algorithm. The idea of early stopping is to maximize the generalization of the network by stopping the training when the prediction performances (in our case, the weighted square error) on the validation dataset begins to rise.

We investigated the optimal value of both neurons in the hidden layer and weight decay coefficient of each specialized predictor by trial and error. The composition of the input

²Available for free from the site www.iau.dtu.dk/research/control/nnsysid.html

(order of autoregressive part and rainfall terms) has been moreover thoroughly analyzed. Given an input set, a network architecture and a value of D , we trained each model 25 times, recording its performance. We finally selected the model which minimizes the criterion J , defined as the sum of the weighted square errors over all the calibration data, i.e. training and validation sets:

$$J = \left[\frac{1}{2N} \sum_{t \in \mathcal{T}} \mu_j(t) [y(t+k) - \hat{y}_j(t+k, \theta)]^2 + \frac{1}{2M} \sum_{t \in \mathcal{V}} \mu_j(t) [y(t+k) - \hat{y}_j(t+k, \theta)]^2 \right] \quad (6)$$

Several authors (see for instance [16]) assume the mean square error evaluated just on the validation set as criterion for selecting the optimal predictor configuration, thus trying to maximize the generalization ability of the model. However, there is no real distinction between the training and the validation sets, which are normally picked up by chance. It is therefore reasonable to adopt criterion (6) for the optimal predictor choice. The effective performances of the models will be finally assessed on the testing set.

5.1 Issuing the forecast

The forecast is issued by weighting the predictors outputs according to the current fuzzy estimation of the state of the basin:

$$\hat{y}(t+k) = \sum_j \mu_j(t) \hat{y}_j(t+k)$$

where index j refers to the different saturation classes. If basin conditions are far from the j -th saturation condition, i.e. $\mu_j = 0$, the j -th predictor does not affect at all the forecast. As the basin conditions become closer to a certain saturation class, higher confidence is smoothly given to the corresponding predictor, avoiding abrupt transitions between models.

6 Results

Judging the effectiveness of a flood forecasting system is a quite complex task. In principle, a correct measure would be a cost-benefit analysis, including the damages from an incorrect forecast (a false or a missing alarm). However, one has always to resort to statistical evaluations since benefits and costs also depends upon a number of external factors such as the way the information is distributed and how people reacts to it. Several indicators of forecast effectiveness have been proposed besides the classical measures of the root mean square error ($RMSE = \frac{1}{N} \sqrt{\sum [y(t) - \hat{y}(t)]^2}$) and correlation ρ between predicted and real flows. For instance, the model efficiency R^2 , defined as:

$$R^2 = \frac{F_0 - F}{F_0} \quad (7)$$

where F_0 is the variance of water levels ($\frac{1}{N} \sum [y(t) - \bar{y}]^2$) and F is the mean square error ($\frac{1}{N} \sum [y(t) - \hat{y}(t)]^2$), is widely used. The “prediction” of the average value, which can be considered as the prediction available even in the worst case, has then an efficiency of 0. An efficiency value of 90% indicates a very satisfactory model performance while a value in the range 80 – 90% indicates a fairly good model [18].

Besides these indicators, which assess the mean performances on the whole dataset, the application claims for a specialized investigation of high flows situations. For example, denoting observed peaks as y_p and predicted ones as \widehat{y}_P , the peak error rate $\frac{y_p - \widehat{y}_P}{y_p}$ [5], or the peak flow criterion $\frac{[\sum (y_p - \widehat{y}_P)^2 y_p^2]^{1/4}}{[\sum \widehat{y}_P^4]^{1/4}}$ [7], are quoted in the literature. However, these indicators do not consider the delays between observed and predicted peaks, which is an essential factor to judge the effectiveness of the model. Here, we use a “high flows error rate” criterion hf , which takes into account all the water levels exceeding the average value μ_y plus twice the square deviation σ_y :

$$hf = E_{(y(t) > \mu_y + 2\sigma_y)} \left[\left| \frac{y(t) - \hat{y}(t)}{y(t)} \right| \right] \quad (8)$$

The number of water levels data exceeding $(\mu_y + 2\sigma_y)$ covers about 2 – 5% of the time series. The indicator would be 0 for a perfect model.

Finally, we characterize the error distribution of the models adopting the following classification:

- class I: error lower than 15 cm (this represents an excellent forecast)
- class II: error between 15 and 30 cm
- class III: error between 30 and 50 cm
- class IV: error higher than 50 cm (this represents a poor forecast).

6.1 Olona case study

The proposed approach has been tested on the case of river Olona (see Figure 4a), located in Lombardia, Northern Italy. The average flow is about $2.5 m^3/sec$, while the maximum expected flow over a time period of 10 years is about $100 m^3/sec$. The area at the considered closing section (Castellanza) sizes about $200 km^2$, and the basin is divided into two parts: the upper one, mountainous and weakly anthropized, and the lower one, flat and strongly urbanized. Besides Castellanza hydrometer, which measures water levels, three rain gauges are available on the basin.

They are located in:

- Arcisate, in the mountainous part of the basin;

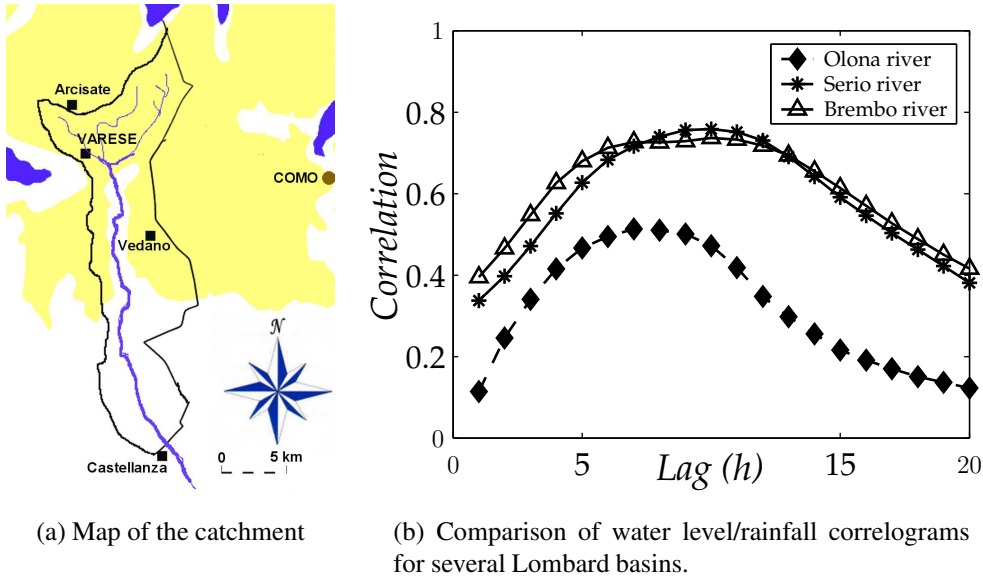


Figure 4: Olona basin characteristics.

- Varese, at the beginning of the urbanized area;
- Vedano, in the urbanized area.

The river receives many artificial inflows in the lower part of the basin and a series of small reservoirs (having an overall capacity of about $208 * 10^3 m^3$), built in order to mitigate floods, alter its natural behavior. Such an heavy anthropization of the lower basin results in decreased values of the cross-correlograms between the time series of hourly water level and of hourly rainfall. This can be easily assessed (Figure 4b) by comparing the cross-correlogramms computed on the most correlated gauge (Arcisate) of the basin with those obtained on other catchments (e.g. Serio and Brembo), located in the same region, and having similar size, but a much less anthropized basin.

Available data refer to 13 events (with an overall length of about 1100 hourly steps) occurred within the period 1999-2001; training, validation and testing sets contain respectively about 540, 400, 140 patterns. Due to the lack of the water level/flow relationship, we adopted the Castellanza water level as variable to be predicted. Water level averages over training, validation and testing sets are respectively 82.3, 83.2, 89.6 cm, while ratios between water level averages and standard deviations are 3.1, 2.3, 2.2. We will evaluate the model performances on a three hours forecast horizon, judged as suitable by Civil Protection technicians.

Two experiments have been performed using the cumulated rainfall computed on two different time windows of 2 and 5 days as saturation proxy; according to the hydrological analysis carried out in Section 3, we expect the 5-days time window to be more effective in assessing the basin state.

A traditional neural network predictor has been identified in order to provide a term of comparison for the performances of the fuzzy framework. The network has an hidden

layer, constituted by neurons having a hyperbolic tangent function, and a unique linear output neuron. Also in this case, we adopt both early stopping and regularization to prevent overfitting. The network architecture, the weight decay coefficient, the input variables set have been tuned by trial and error, minimizing the sum of square error over training and validation sets, coherently with the optimality criterion (6) formulated for the specialized predictors.

6.2 Forecast performances

We present the results obtained by modelling the basin state through two fuzzy saturation classes, since using three or more classes does not result in improved performances. Centroids have been identified at 25 and 100 mm on the 2-days cumulated rainfall, and are about 15-20% higher in the 5-days case (31 and 114 mm respectively).

Model	hidden nodes	regression order
<i>traditional ANN</i>	3	3
<i>fuzzy framework</i> ($\Psi = 2$ days)		
<i>network 1</i> (“dry basin“)	3	4
<i>network 2</i> (“wet basin“)	9	4
<i>fuzzy framework</i> ($\Psi = 5$ days)		
<i>network 1</i> (“dry basin“)	3	2
<i>network 2</i> (“wet basin“)	9	2

Table 1: Models architectures.

	$\Psi = 2$ days		$\Psi = 5$ days	
	$\sum \mu_1(t)$	$\sum \mu_2(t)$	$\sum \mu_1(t)$	$\sum \mu_2(t)$
<i>training</i> (\mathcal{T})	308	235	319	224
<i>validation</i> (\mathcal{V})	232	165	199	198
<i>testing</i> (\mathcal{S})	69	73	70	72

Table 2: Sums of classes memberships on the three sets.

In Table 1 we present the architectural features of the selected networks: a *regression order 2*, for instance, means that the input contains 2 autoregressive terms, $y(t-1)$, $y(t-2)$, and 2 terms, $r_i(t-1)$, $r_i(t-2)$, for each rain gauge i .

It is interesting to notice that the networks trained on the second saturation class contain more hidden nodes than the first ones, trained on dryer conditions, and that therefore the second class comprises most of the non-linearities of the process. These empirical findings seem to suggest that the rainfall-runoff relationship which holds on the wetter class is more complex than that on the dryer class. Since hydrological studies [24] have shown that as the catchment becomes saturated the soil tends to behave as an impervious surface, thus simplifying the rainfall-runoff relationship, we hypothesize that many data

belonging at some extent to the second class do not correspond to completely saturated conditions.

On the other hand, the “classical” neural network tends to capture a somehow “average” basin saturation state. One can suppose that it will contain more neurons as the number of data in the time series belonging to the second saturation class increases, and vice-versa. The sum of memberships μ_1 and μ_2 along the time series provides a measure of the prevailing saturation condition, as estimated by our proxy variable. The prevalence of data belonging to the “dry” class in the calibration time series (see Table 2) may therefore explain why the ANN predictor structure tends to be similar to the specialized model identified for the first class.

Model	training				testing			
	R^2	ρ	RMSE	hf	R^2	ρ	RMSE	hf
ANN	.910	.954	9.72	.116	.853	.933	15.37	.162
fuzzy framework ($\Psi = 2$ days)	.946	.973	7.87	.008	.857	.941	15.18	.196
fuzzy framework ($\Psi = 5$ days)	.925	.963	8.20	.109	.879	.948	13.95	.150

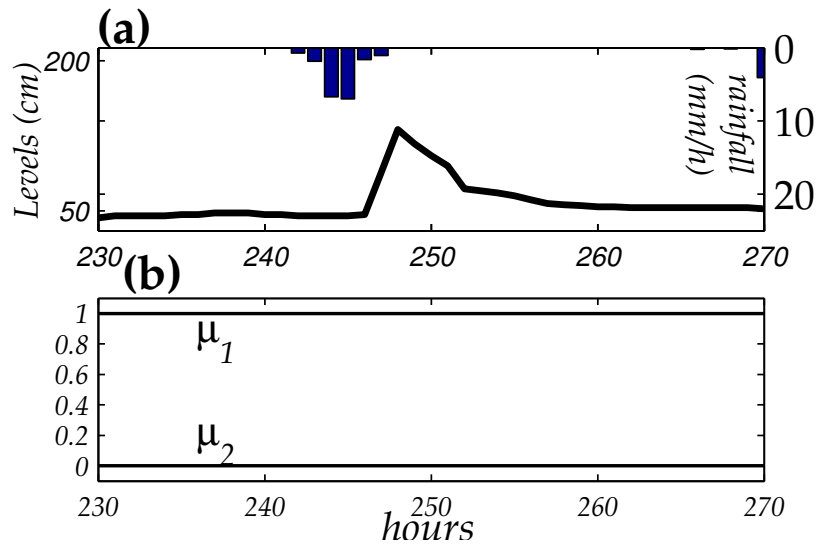
Table 3: 3-hours ahead forecast performances.

Model / Error class	training (536 data)				testing (392 data)			
	I	II	III	IV	I	II	III	IV
ANN	504	28	2	2	324	40	22	6
fuzzy framework ($\Psi = 2$ days)	528	7	0	1	307	56	25	4
fuzzy framework ($\Psi = 5$ days)	514	21	0	1	324	46	18	4

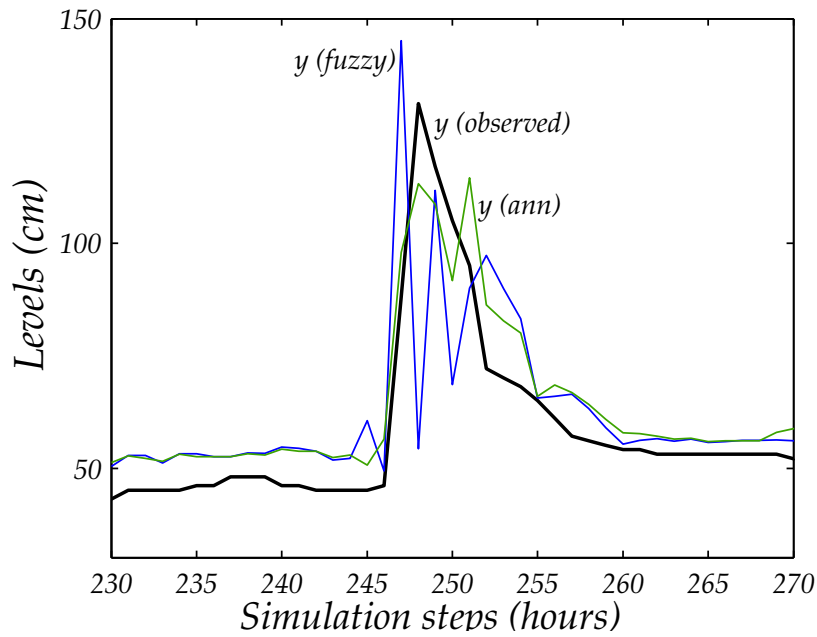
Table 4: Error distributions for the different models.

Table 3 shows the models performances over training and testing sets. As expected because of the greater number of parameters, both the fuzzy frameworks overperform the classical ANN over the training set; they perform better also on the testing set, when they run on previously unseen data. Using the 5-days cumulated rainfall allows the highest performances, thus confirming the suitability of such an information as proxy for the basin state. The error distribution of the models, reported in Table 4, confirm the improvement of the fuzzy framework having $\Psi = 5$ days with respect to the other cases.

From a physical point of view, the prediction improvement provided by the proposed framework over the classical ANN approach is bounded by changes in the basin saturation which may occur during the k steps of the forecast horizon: indeed, if rainfall continues during these k steps, the basin state is likely to change, and this cannot be captured by the current memberships values $\mu(t)$. Such a behavior is clearly shown by figure 5 and 6, which present simulation samples taken from the testing set, using the 5-days cumulated rainfall information. During the event plotted in figure 5, the basin state is constant, with $[\mu_1 = 1, \mu_2 = 0]$, because of low rainfall; therefore, the framework uses always the “dry”

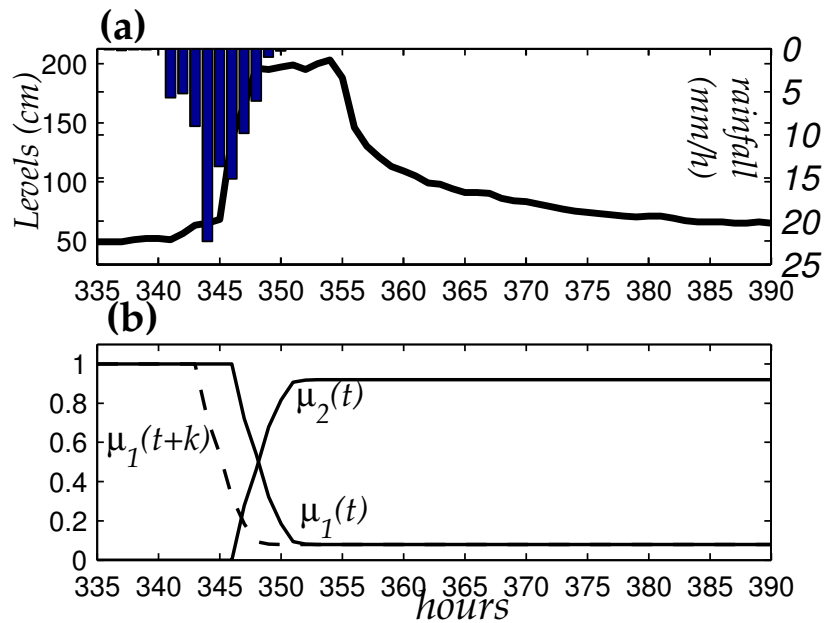


(a) Upper plot: average hourly rainfall on the basin (reversed axis) and water levels. Lower plot: fuzzy memberships of the basin saturation state.

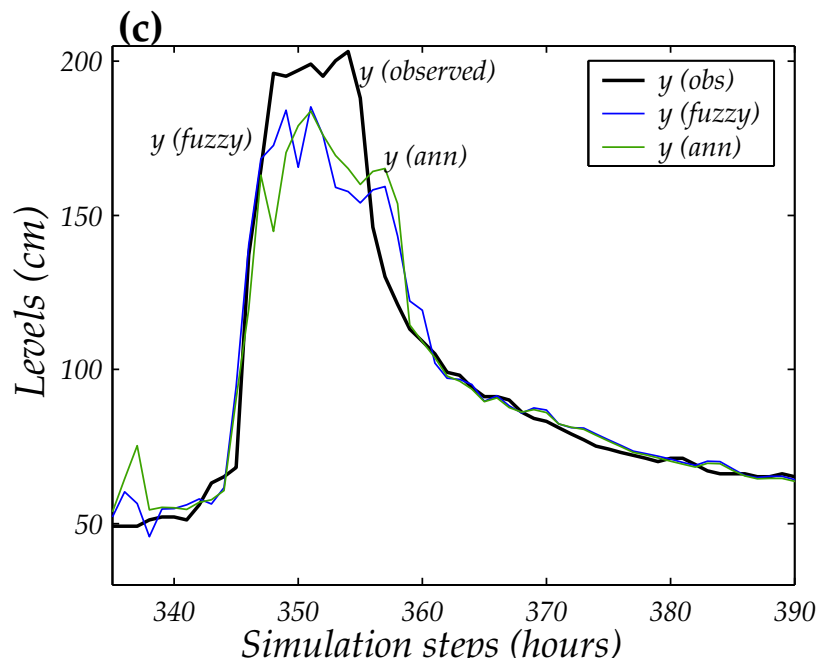


(b) Simulations

Figure 5: 3-hours prediction samples (testing set). The forecast issued by the specialized models are denoted by \hat{y}_1 and \hat{y}_2 , their fuzzy combination by \hat{y} , and the ANN forecast by \widehat{ann} . In the fuzzy simulation, the forecast of the first network \hat{y}_1 is exactly covered by the framework forecast \hat{y} , since $\mu_1 = 1$ at each time step.



(a) Upper plot: average hourly rainfall on the basin (reversed axis) and water levels. Lower plot: fuzzy memberships of the basin saturation state.



(b) Simulations

Figure 6: 3-hours prediction samples (testing set). The forecast issued by the specialized models are denoted by \hat{y}_1 and \hat{y}_2 , their fuzzy combination by \hat{y} , and the ANN forecast by \hat{ann} .

predictor. providing a better accuracy than the classical ANN, which can represent only an “average” basin state.

In figure 6, we show a further flood event, characterized by much higher rainfall and consequently greater flows. Due to the high rainfall values, the saturation changes which occur within a three hours time window are relevant; this causes a discrepancy between the basin saturation at time (t), when the forecast is computed, and its actual value at time ($t+3$), which is the time horizon the forecast refers to. For example, one can notice that, around the peak flow, the prediction $\hat{y}(t+3)$ takes still into consideration the prediction issued by the “dry” network. Nevertheless, even in this unfavorable case, where significant discrepancies between $\mu_i(t)$ and $\mu_i(t+k)$ are observed, the fuzzy framework does not behave worse than the classical ANN.

7 Conclusions

Neural networks are increasingly used in hydrological research, given their ability in capturing the non linearities involved in the rainfall-runoff relationship. This paper focuses on the improvement of the representation of the basin saturation state, which is usually described just in terms of past flow measurements. The basin state is classified through a fuzzy approach which analyzes the cumulated rainfall information; for each saturation class, a specialized neural predictor is identified. The forecast is finally obtained by weighting the predictions of the local models according to the current basin state fuzzy classification; the fuzzy approach avoids abrupt transition between the different predictors.

Performance improvements have been found by comparing the proposed framework approach with respect to traditional ANN. This progress is due to the combined action of two factors: one is the exploitation of the novel information (the cumulated rainfall), which does not normally appears in ANN flood predictors, and the other is the fuzzy approach in which such an information has been used.

A future subject of investigation may be the development of a novel algorithm able to estimate in an integrated way both the membership functions and the parameters of the specialized predictors.

Acknowledgments. *The authors thank the Civil Protection Service of Regione Lombardia for supplying the data of the Olona river.*

Appendix A - Weighted least square training algorithm

A description of the Levenberg Marquardt algorithm for weighted least square is given in [17]; here we present how it has been applied for the training of a feed forward neural network, like that presented in Section 2. To simplify the notation we refer to a given specialized model, denoting its error as $\epsilon(t)$, without introducing a specific index for the saturation class.

On the basis of equation (3), the derivatives of $\epsilon(t)$ with respect to the output neuron weights W_i and bias b_{out} are given by:

$$\frac{\partial \epsilon(t)}{\partial W_i} = -c_i \quad \frac{\partial \epsilon(t)}{\partial b_{out}} = -1 \quad (9)$$

Error derivatives with respect to weights w_{ij} and biases b_j in the hidden layer are obtained via backpropagation: observing that $\epsilon(t)$ depends on w_{ij} and σ_j only via the summed input z_j to the hidden node j , one can write:

$$\frac{\partial \epsilon(t)}{\partial w_{ij}} = \frac{\partial \epsilon(t)}{\partial z_j} \frac{\partial z_j}{\partial w_{ij}} \quad \frac{\partial \epsilon(t)}{\partial b_j} = \frac{\partial \epsilon(t)}{\partial z_j} \frac{\partial z_j}{\partial b_j} \quad (10)$$

and considering that \hat{y} depends on z_j only via the output c_j of the j -th hidden node:

$$\frac{\partial \epsilon(t)}{\partial z_j} = \frac{\partial \epsilon(t)}{\partial c_j} \frac{\partial c_j}{\partial z_j} = -W_j f'(z_j) \quad (11)$$

Substituting into (10), we obtain:

$$\frac{\partial \epsilon(t)}{\partial w_{ij}} = -W_j f'(z_j) u_i \quad \frac{\partial \epsilon(t)}{\partial b_j} = -W_j f'(z_j) \quad (12)$$

Formulas (9, 12) provide the derivatives of the unweighted error $\epsilon(t)$ with respect to all the network parameters. Now, let us consider the weighted objective function:

$$\Phi = \frac{1}{2N} \sum_{t=1}^N \mu(t) \epsilon(t)^2 \quad (13)$$

The gradient G of this function is a vector of the same size of θ , whose entries are given by:

$$G_i = \frac{\partial \Phi}{\partial \theta_i} = \frac{1}{N} \sum_{t=1}^N \mu(t) \epsilon(t) \frac{\partial \epsilon(t)}{\partial \theta_i} \quad (14)$$

while the Hessian H is a square matrix, whose entries are given by:

$$\begin{aligned}
H_{ij} &= \frac{\partial^2 \Phi}{\partial \theta_i \partial \theta_j} = \frac{1}{N} \sum_{t=1}^N \mu(t) \frac{\partial \epsilon(t)}{\partial \theta_j} \frac{\partial \epsilon(t)}{\partial \theta_i} + \frac{1}{N} \sum_{t=1}^N \mu(t) \frac{\partial^2 \epsilon(t, \theta)}{\partial \theta_i \partial \theta_j} \epsilon(t) \cong \\
&\cong \frac{1}{N} \sum_{t=1}^N \mu(t) \frac{\partial \epsilon(t)}{\partial \theta_j} \frac{\partial \epsilon(t)}{\partial \theta_i} \quad (15)
\end{aligned}$$

The second summation, which is clearly zero in the case of a linear network, can be ignored even in the non linear case, since the terms $\epsilon(t)$ multiplying the second derivative should be just a random error uncorrelated with the model, thus tending to cancel the second derivative when summed over t [17]. This approximation allows an efficient Hessian evaluation. Let us introduce the matrices J and \tilde{J} such that:

$$J = \begin{bmatrix} \frac{\partial \epsilon(1)}{\partial \theta_1} & \frac{\partial \epsilon(1)}{\partial \theta_2} & \cdots & \frac{\partial \epsilon(1)}{\partial \theta_n} \\ \frac{\partial \epsilon(2)}{\partial \theta_1} & \frac{\partial \epsilon(2)}{\partial \theta_2} & \cdots & \frac{\partial \epsilon(2)}{\partial \theta_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \epsilon(N)}{\partial \theta_1} & \frac{\partial \epsilon(N)}{\partial \theta_2} & \cdots & \frac{\partial \epsilon(N)}{\partial \theta_n} \end{bmatrix} \quad (16)$$

$$\tilde{J} = \begin{bmatrix} \mu(1) \frac{\partial \epsilon(1)}{\partial \theta_1} & \mu(1) \frac{\partial \epsilon(1)}{\partial \theta_2} & \cdots & \mu(1) \frac{\partial \epsilon(1)}{\partial \theta_n} \\ \mu(2) \frac{\partial \epsilon(2)}{\partial \theta_1} & \mu(2) \frac{\partial \epsilon(2)}{\partial \theta_2} & \cdots & \mu(2) \frac{\partial \epsilon(2)}{\partial \theta_n} \\ \vdots & \vdots & \ddots & \vdots \\ \mu(N) \frac{\partial \epsilon(N)}{\partial \theta_1} & \mu(N) \frac{\partial \epsilon(N)}{\partial \theta_2} & \cdots & \mu(N) \frac{\partial \epsilon(N)}{\partial \theta_n} \end{bmatrix} \quad (17)$$

Since coefficients $\mu(t)$ do not depend on θ , the partial derivatives in J and \tilde{J} can be actually computed by equations (9, 12) obtained for the unweighted case.

Denoting by $\bar{\epsilon}$ the error vector, the gradient G and Hessian H can be obtained as:

$$G = \tilde{J} \bar{\epsilon} \quad (18)$$

$$H = \tilde{J} J^T \quad (19)$$

where we approximated the Hessian as explained above .

The Levenberg Marquardt rule for parameters update, which is defined in the classical least square case as:

$$\theta^{i+1} = \theta^i - [J J^T + \lambda I]^{-1} J^T \bar{\epsilon}(\theta^i) \quad (20)$$

becomes finally:

$$\theta^{i+1} = \theta^i - [\tilde{J} J^T + \lambda I]^{-1} \tilde{J} \bar{\epsilon}(\theta^i) = \theta^i - [H + \lambda I]^{-1} G^T \bar{\epsilon}(\theta^i) \quad (21)$$

for the weighted least square criterion. If we want to regularize the training function as:

$$\Phi_r = \frac{1}{2N} \sum_{t=1}^N \mu(t) \epsilon(t)^2 + D \|\theta\|^2 \quad (22)$$

the gradient G_r and Hessian H_r to be used in the update rule become respectively:

$$G_r = G + D\theta \quad (23)$$

$$H_r = H + ID \quad (24)$$

References

- [1] A.F. Atiya, S.M. El-Shoura, I.S. Shaheen, and M.S. El-Sherif. A comparison between neural-network forecasting techniques case study: River flow forecasting. *IEEE Trans. Neural Networks*, 10(2):402–409, 1999.
- [2] J.C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, 1981.
- [3] C.M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.
- [4] M. Campolo, P. Andreussi, and A. Soldati. River flood forecasting with a neural network model. *Water Resour. Res.*, 35(4):1191–1197, 1999.
- [5] F.J. Chang, J.M. Liang, and Y.C. Chen. Flood forecasting using radial basis function neural networks. *IEEE Trans. Syst., Man, Cybern. C*, 31(4):530–535, 2001.
- [6] H. Elragal and A. Khotanzad. Natural gas load forecasting with combination of adaptive neural networks. In *Proceedings of International Joint Conference on Neural Networks (IJCNN'99)*, pages 4069–4072, 1999.
- [7] R. Garcia Bartual. Short term river flood forecasting with neural networks. In A.E. Rizzoli and A.J. Jakeman, editors, *iEMSs 2002 International Congress: Integrated Assesment and Decision Support*, volume 2, pages 160–165, 2002.
- [8] F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural networks architectures. *Neural Computation*, 7:219–269, 1995.
- [9] S. Hashem and B. Schmeiser. Improving model accuracy using optimal linear combinations of trained neural networks. *IEEE Trans. Neural Networks*, 6(3):792–794, 1995.
- [10] R.K. Kachroo and L. Natale. Non linear modelling of the rainfall runoff tranformation. *J. of Hydrology*, 133:41–97, 1992.
- [11] G. Kim and A.P Barros. Quantitative flood forecasting using multisensor data and neural networks. *J. of Hydrology*, 246:45–62, 2001.
- [12] W. Krajewski, V. Lakshmi, K. Georgakakos, and S.C. Jahn. A Monte Carlo study of rainfall sampling effect on a distributed catchment model. *Water Resour. Res.*, 27:119–128, 1991.
- [13] Y. Le Cun, I. Kanter, and S. Solla. Eigenvalues of covariance matrices: application to neural network learning. *Phys. Rev. Letters*, 14(1):2396–2399, 1991.
- [14] H.R. Maier and C.G. Dandy. Neural networks for the prediction and forecasting of water resources variables: a review of modelling issues and applications. *Environmental Mod. & Soft.*, 15:101–124, 2000.

- [15] C.S. Melching, B.C. Yen, and H.G. Wenzel. A reliability estimation in modeling watershed runoff with uncertainties. *Water Resour. Res.*, 26(10):2275–2286, 1990.
- [16] M. Norgaard, O. Ravn, N.K. Poulsen, and L.K. Hansen. *Neural Networks for Modelling and Control of Dynamic Systems*. Springer-Verlag, London, 2000.
- [17] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C*. Cambridge University Press, 1988.
- [18] A.Y. Shamseldin, K.M. O’Connor, and K.M. Liang. Methods for combining the outputs of different rainfall-runoff models. *J. of Hydrology*, 197:203–229, 1997.
- [19] V.P. Singh and D.A. Woolhiser. Mathematical modeling of watershed hydrology. *J. of Hydrologic Engineering*, 7(4):270–292, 2002.
- [20] E. Todini. The ARNO rainfall-runoff model. *J. of Hydrology*, 175:339–382, 1996.
- [21] E. Todini and J.R. Wallis. Using cfs for daily or longer period rainfall runoff modelling. In T.A. Ciriani, U. Maione, and J.R. Wallis, editors, *Mathematical models for surface water hydrology*, 1977.
- [22] S. Tokar and M. Markus. Precipitation-runoff modeling using artificial neural networks and conceptual models. *J. of Hydrologic Engineering*, 5(2):156–161, 2000.
- [23] H. Tong. *Threshold Models in Non-linear Time Series Analysis*. Springer Verlag, 1983.
- [24] D.A. Woolhiser and J.A. Liggett. Unsteady, one-dimensional flow over a plane -the rising hydrograph. *Water Resour. Res.*, 3(3):753–771, 1967.
- [25] L. Xiong, A.Y. Shamseldin, and K.M. O’Connor. A non linear combination of the forecasts of rainfall runoff models by the first order Takagi Sugeno fuzzy system. *J. of Hydrology*, 245:196–217, 2001.
- [26] J. Zhang and A.J. Morris. Fuzzy neural networks for nonlinear systems modelling. *IEE Proc.-Control Theory Appl.*, 142(6):551–561, 1995.

List of Figures

1	Structure of a fully connected neural network.	5
2	Sample of membership functions with three centroids.	7
3	λ as a function of cumulated rainfalls at the beginning of the floods. Dashed lines show the linear interpolations.	8
4	Olona basin characteristics.	12
5	3-hours prediction samples (testing set). The forecast issued by the specialized models are denoted by \hat{y}_1 and \hat{y}_2 , their fuzzy combination by \hat{y} , and the ANN forecast by \widehat{ann} . In the fuzzy simulation, the forecast of the first network \hat{y}_1 is exactly covered by the framework forecast \hat{y} , since $\mu_1 = 1$ at each time step.	15
6	3-hours prediction samples (testing set). The forecast issued by the specialized models are denoted by \hat{y}_1 and \hat{y}_2 , their fuzzy combination by \hat{y} , and the ANN forecast by \widehat{ann}	16

List of footnotes

Manuscript received

Giorgio Corani, Giorgio Guariso

Politecnico di Milano

Dipartimento di Elettronica ed Informazione

Via Ponzio, 34/5 - 20133 MILANO (Italy)

1. The Thiessen method subdivides the catchment area into several polygons, whose boundaries are located at a distance halfway between gauge pairs. In practice, the rainfall is then assumed to be constant over each polygon. $R(\Psi, t)$ is finally estimated as:

$$R(\Psi, t) = \sum_{\tilde{t}=t-\Psi}^{\tilde{t}=t} \sum_{i=1}^{i=m} s_i r_i(\tilde{t}) \quad (25)$$

where s_i is the ratio of the area of the i -th polygon to the whole catchment. *Page 6*

1. Available for free from the site www.iau.dtu.dk/research/control/nnsysid.html
Page 9