Chapter

Nonlinear Evapotranspiration Modeling Using Artificial Neural Networks

Sirisha Adamala

Abstract

Reference evapotranspiration (ET_o) is an important and one of the most difficult components of the hydrologic cycle to quantify accurately. Estimation/measurement of ET_o is not simple as there are number of climatic parameters that can affect the process. There exists copious conventional (direct and indirect) and non conventional/soft computing (artificial neural networks, ANNs) methods for estimating ET_o. Direct methods have the limitations of measurement errors, expensive, impracticality of acquiring point measurements for spatially variable locations, whereas the indirect methods have the limitations of unavailability of all necessary climate data and lack of generalizability (needs local calibration). In contrast to conventional methods, soft computing models can estimate ET_o accurately with minimum climate data which have advantages over limitations of conventional ET_o methods. This chapter reviews the application of ANN methods in estimating ET_o accurately for 15 locations in India using six climatic variables as input. The performance of ANN models were compared with the multiple linear regression (MLR) models in terms of root mean squared error, coefficient of determination and ratio of average output and target ET_o values. The results suggested that the ANN models performed better as compared to MLR for all locations.

Keywords: evapotranspiration, ANN, climate, data, Gaussian, lysimeter

1. Introduction

Evapotranspiration (ET) is the combining process of evaporation and transpiration losses. Almost 62% of precipitation falls on continents are returned back to the atmosphere through the ET process [1]. ET plays a significant role in the hydrological cycle and its estimation is very important in various fields of water resources. A common procedure for estimating actual crop evapotranspiration (ET_{crop}) is to first estimate reference evapotranspiration (ET_o) and to then apply an appropriate crop coefficient (k_c). ET_o is an important and one of the most difficult components of the hydrologic cycle to quantify accurately. ET_o is measured from a hypothetic crop of uniform height (12 cm), active growing (crop resistance of 70 s m^{-1}), completely shading the ground (albedo of 0.23) and unlimited supply of water [2]. The Food and Agricultural Organization (FAO) consider the above definition as standard and sole method for estimating ET_o if sufficient climatic data are available [3, 4].

Estimation of ET_o is complex due to influence of various climatic variables (maximum and minimum air temperature, wind speed, solar radiation and maximum and minimum relative humidity) and existence of nonlinearity in between climatic data and ET_o. Though users have number of methods for measuring or estimating ET_o directly or indirectly, most of them have some limitations regarding data availability or regional applicability. In addition, in order to use these methods, users are required to make reasonable estimates for some of the parameters in the employed ET_o models, which involve some uncertainties and might not result in reliable ET_o estimates [5]. Further, it is difficult to develop accurately representative physically based models for the complex non-linear hydrological processes, such as ET_o. This is because the physical relationships involving in a system can be too complicated to be accurately represented in a physically based manner.

The above limitations lead to the need of developing some techniques that can accurately estimate ET_o values with a minimum input data and are also easy to apply without knowing physical process inside the system. Artificial neural network (ANN) techniques, which can provide a model to predict and investigate the process without having a complete understanding of it, can be a useful tool for the above purpose. These techniques are also interesting because of its knowledge discovering property. In contrast to conventional methods, ANNs can estimate ET_o accurately with minimum climate data, which may have the advantages of being inexpensive, independent of specific climatic condition, ignored physical relations, and precise modeling of nonlinear complex system. In the last decade, many researchers have used ANN techniques for modeling of the ET_o process [6–25].

2. Review of literature

This section follows the discussion of some of the significant contributions made by various researchers in the application of different ANN techniques for modeling ET_o or pan evaporation (E_p). A radial basis function (RBF) neural network was developed in C language to estimate daily soil water evaporation [26]. The input layer of network consists of average relative air humidity, air temperature, wind speed (Ws) and soil water content. They compared the results of RBF networks with the multiple linear regression (MLR) techniques. A feed-forward back propagation (FFBP)-based ANN model was developed to estimate daily E_p based on measured weather variables [27]. They used different input combinations to model E_p. They compared the developed ANN models with the Priestly-Taylor & MLR models. RBF neural network model was developed to estimate the FAO Blaney-Criddle *b* factor [28]. The input layer to RBF model consists of minimum daily relative humidity (RH_{min}), day time Ws and mean ratio of actual to possible sunshine hours (n/N). The *b* values estimated by the RBF models were compared to the appropriate *b* values produced using regression equations. FFBP ANN models were implemented for the estimation of daily ET_o using six basic climatic parameters as inputs [16]. They trained ANNs using three learning methods (with different learning rates and momentum coefficients), different number of processing elements in the hidden layers, and the number of hidden layers. The compared the results of developed ANN models with the Penman Monteith (PM) method and with a lysimeter measured ET_o. ANN-based back propagation models for estimating Class A E_p with minimum climate data (four input combinations) were developed and compared with the existing conventional methods [22].

The potential of RBF neural network for estimating the rice daily crop ET using limited climatic data was demonstrated [23]. Six RBF networks, each using varied input combinations of climatic variables were trained and tested. The model estimates were compared with measured lysimeter ET. A sequentially adaptive RBF network was applied for forecasting of monthly ET_o [29]. Sequential adaptation of parameters and structure was achieved using extended Kalman filter. Criterion for network growing was obtained from the Kalman filter's consistency test. Criteria for neuron/connections pruning were based on the statistical parameter significance test. The results showed that the developed network was learned to forecast $ET_{o,t+1}$ (current or next month) based on $ET_{o,t-11}$ (at a lag of 12 months) and $ET_{o,t-23}$ (at a lag of 24 months) with high reliability. The study examined that whether it is possible to attain the reliable estimation of ET_o only on the basis of the temperature data [24]. He developed RBF neural network for estimating ET_o and compared the developed model with temperature-based empirical models.

ANN-based daily ET_o models were trained based on different categories of conventional ET_o estimation methods such as temperature based (FAO-24 Blaney-Criddle), radiation based (FAO-24 Radiation method for arid and Turc method for humid regions) and combinations of these (FAO-56 PM) [14]. A comparison of the Hargreaves and ANN methods for estimating monthly ET_o only on the basis of temperature data was done [19]. They developed ANN models with the data from six stations and tested these developed models with the data from remaining six stations, which were not used in model development. The capability of ANN for converting E_p to ET_o was studied using temperature data [18]. The conventional method that uses pan coefficient (K_p) as a factor to convert E_p to ET_o was considered for this comparison. Generalized ANN (GANN)-based ET_o models corresponding to FAO-56 PM, FAO-24 Radiation, Turc and FAO-24 Blaney-Criddle methods were developed [15]. These models were trained using the pooled data from four California Irrigation Management Information System (CIMIS) stations with FAO-56 PM computed values as targets. The developed GANN models were tested with different stations which were not used in training. Multilayer perceptron (MLP) neural networks for estimating the daily E_p using input data of maximum and minimum air temperature and the extraterrestrial radiation was developed [20]. The potential for the use of ANNs to estimate the ET_o based on air temperature data was examined [21]. He also conducted comparison of estimates provided by the ANNs and by Hargreaves equation by using the FAO-56 PM model as reference model.

3. Study area and data collected

For the purpose of this study, 15 meteorological stations in India were chosen. **Figure 1** shows the geographical locations of these selected stations and their related agro-ecological regions (AERs). These stations are having daily meteorological data of from 2001 to 2005 of following variables: minimum air temperature (T_{min}), maximum air temperature (T_{max}), minimum relative humidity (RH_{min}), maximum relative humidity (RH_{max}), mean wind speed (w_s), and solar radiation (S_{ra}). **Table 1** shows the details of 15 climatic stations of India along with altitude and duration of available data. The study area is bounded between the longitudes of 68° 7′ and 97° 25′ E and the latitudes of 8° 4′ and 37° 6’ N. The annual potential evapotranspiration of India is 1771 mm. It varies from a minimum of 1239 mm in Jammu and Kashmir to a maximum of 2100 mm in Gujarat [30].

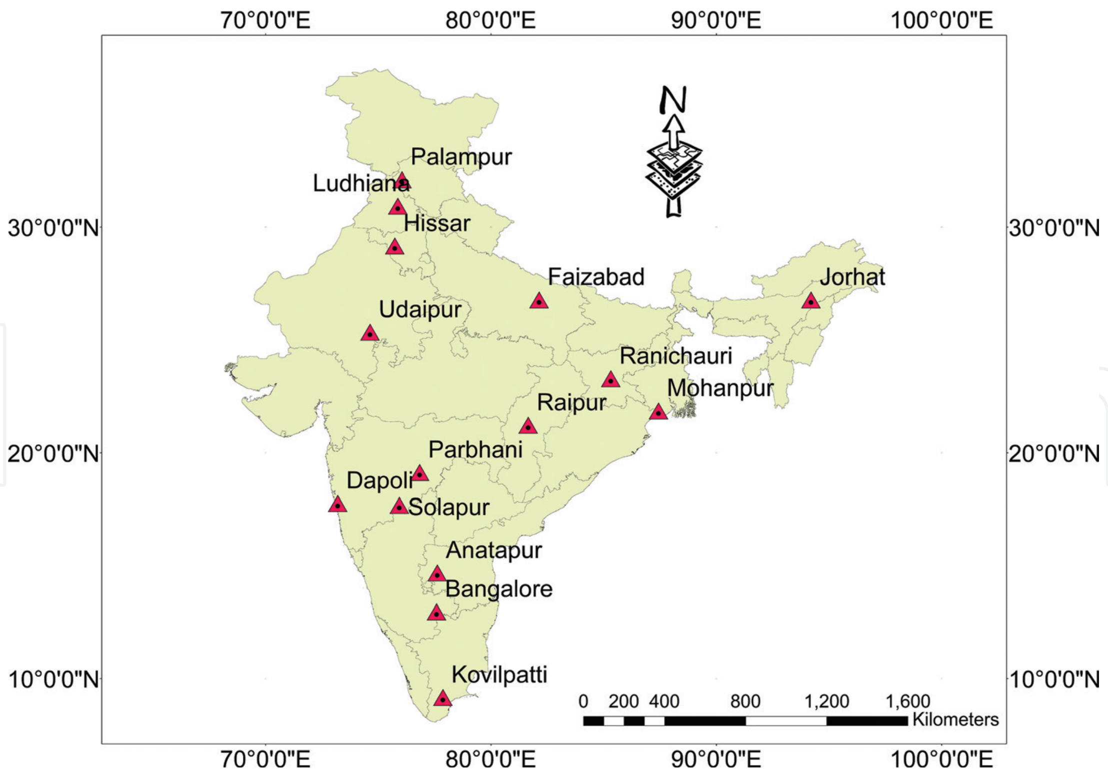

Figure 1.
Geographical locations of study sites in India.

AER	Location	Alt. (m)	Period	T_{max} (°C)	T_{min} (°C)	RH_{max} (%)	RH_{min} (%)	W_s (km h^{-1})	S_{ra} (MJ m^{-2} day^{-1})
Semi-arid	Parbhani	423	2001–2005	33.75	18.32	71.13	41.02	5.04	20.87
	Solapur	25	2001–2005	34.15	20.14	73.28	45.09	6.15	18.96
	Bangalore	930	2001–2005	28.90	17.70	89.15	47.30	8.68	18.95
	Kovilpatti	90	2001–2005	35.11	23.37	80.36	48.52	6.60	19.30
	Udaipur	433	2001–2005	31.81	16.33	72.36	36.44	3.74	19.45
Arid	Anantapur	350	2001–2005	34.43	21.78	73.32	33.91	9.64	20.27
	Hissar	215	2001–2005	31.17	16.23	81.00	44.27	5.20	17.26
Sub-humid	Raipur	298	2001–2005	32.60	19.91	80.62	44.08	5.33	17.80
	Faizabad	133	2001–2005	31.56	18.18	87.02	52.11	3.51	17.88
	Ludhiana	247	2001–2005	30.06	17.42	83.97	49.14	4.26	18.10
	Ranichauri	1600	2001–2005	20.08	9.66	81.15	61.55	4.99	16.23
Humid	Palampur	1291	2001–2005	24.41	13.24	69.70	57.88	5.56	16.35
	Jorhat	86	2001–2005	27.97	19.23	92.70	75.27	3.00	14.68
	Mohanpur	10	2001–2005	32.20	21.04	96.18	61.48	1.27	18.06
	Dapoli	250	2001–2005	31.13	18.87	93.77	69.22	4.92	18.02

Table 1.
Station locations and period of records.

4. Theoretical consideration

The concept of neural networks was introduced by [31]. The neural-network approach, also referred to as 'connectionism' or 'paralleled distributed processing', adopts a "Brain metaphor" of information processing. Information processing in a neural network occurs through interactions involving large number of simulated neurons. A neural network (NN) is a simplified model of the human brain nervous system consisting of interconnected neurons in a parallel distributed system, which can learn and memorize the information. In NN, the interneuron connection strengths, known as 'synaptic weights' are used to store the acquired knowledge [32]. In other words, ANN discovers the relationship between a set of inputs and desired outputs without giving any information about the actual processes involved; it is in essence based on pattern recognition. ANNs consist of a number of interconnected processing elements or neurons. How the inter-neuron connections are arranged determines the topology of a network. A neuron is the fundamental unit of human brain's nervous system that receives and combines signals from other neurons through input paths called 'dendrites'. Each signal coming to a neuron along a dendrite passes through a junction called 'synapse', which is filled with neuro-transmitter fluid that produce electrical signals to reach to the soma or cell body where processing of the signals occurs [16]. If the combined input signal after processing is stronger than the threshold value, the neuron activates, producing an output signal, which is transferred through the axon to the other neurons. Similarly, ANN consists of a large number of simple processing units called neurons (or nodes) linked by weighted connections. A comprehensive description of neural networks was presented in a series of papers [33–35], which provide valuable information for the researchers.

4.1 Model of a neuron

The main function of artificial neuron is to generate output from an activated nonlinear function with the weighted sum of all inputs. **Figure 2** illustrates a nonlinear model of a neuron, which forms the basis for designing ANN. The input layer neurons receive the input signals (x_i) and these signals are passed to the cell body through the synapses. A set of synapses or connecting links is characterized by its own weight or strength. A signal at the input of synapse 'i' connected to neuron 'k' is multiplied by the synaptic weight 'w_{ki}'. The input signals, weighted by the respective synapses of the neuron are added by a linear combiner. An activation function or squashing function is used for limiting the permissible amplitude range of the output of a neuron to some finite value. An external bias (b_k) has an effect of increasing or decreasing the net input of the activation function depending on whether it is positive or negative, respectively.

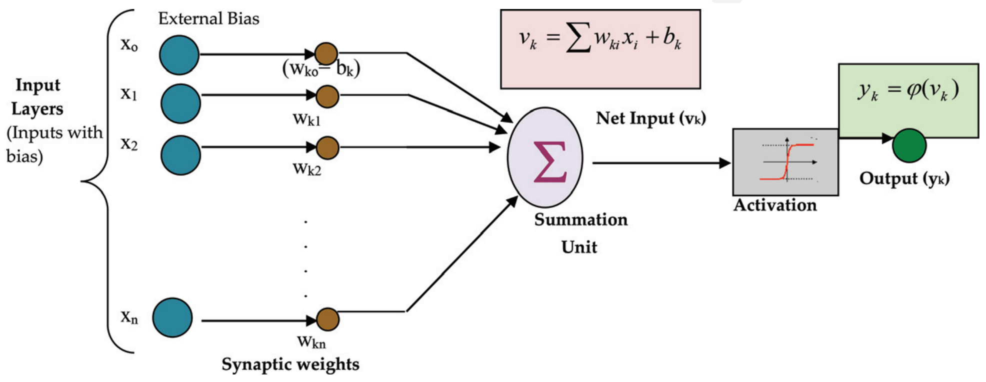

Figure 2.
A nonlinear model of a neuron.

In the mathematical form, a neuron k may be described by the following equations:

$$u_k = \sum_{i=1}^{n} w_{ki} x_i \tag{1}$$

$$yk = \phi(uk + bk) \tag{2}$$

where x_1, x_2, x_3, ... x_n = input signals; w_{k1},w_{k2}, ...w_{kn} = synaptic weights of neuron k; u_k = linear combiner output due to the input signal; b_k = bias; $\varphi(.)$ = activation function; y_k = output signal of the neuron k.

Let v_k be the induced local field or activation potential, which is given as:

$$v_k = u_k + b_k \tag{3}$$

Now, Eqs. (1), (2) and (3) can be written as:

$$v_k = \sum_{i=0}^{m} w_{kn} x_n \tag{4}$$

$$yk = \phi(vk) \tag{5}$$

In Eq. (5), a new synapse with input x_0 = +1 is added and its weight is w_{k0} = b_k to consider the effect of the bias.

4.2 Neural network architecture parameters

Determination of appropriate neural network architecture is one of the most important tasks in model-building process. Various types of neural networks are analyzed to find the most appropriate architecture of a particular problem. Multilayer feed forward networks are found to outperform all the others. Although multilayer feed forward networks are one of the most fundamental models, they are the most popular type of ANN structure suited for practical applications.

4.3 Number of hidden layers

There is no fixed rule for selection of hidden layers of a network. Therefore, trial and error method was used for selection of number of hidden layers. Even one hidden layer of neuron (operating sigmoid activation function) can also be sufficient to model any solution surface of practical interest [36].

4.4 Number of hidden neurons

The ability of the ANN to generalize data not included in training depends on selection of sufficient number of hidden neurons to provide a means for storing higher order relationships necessary for adequately abstracting the process. There is no direct and precise way of determining the most appropriate number of neurons to include in hidden layer and this problem becomes more complicated as number of hidden layer increases. Some studies indicated that more number of neurons in hidden layer provide a solution surface that closely fit to training patterns. But in practice, more number of hidden neurons results the solution surface that deviate significantly from the trend of the surface at intermediate points or provide too literal interpretation of the training points which is called ‘over fitting’. Further, large number of hidden neurons reduces the speed of operation of network during training and testing. However, few hidden neurons results inaccurate model and provide a solution surface that deviates from training patterns. Therefore, choosing optimum number of hidden neurons is one of the important training parameter in ANN. To solve this problem, several neural networks with different number of hidden neurons are used for calibration/training and one with best performance together with compact structure is accepted.

4.5 Types of activation functions

The activation function or transfer function, denoted by $\varphi(v)$, defines the output of a neuron in terms of the induced local field v. It is valuable in ANN applications as it introduces a degree of nonlinearity between inputs and outputs. Logistic sigmoid, hyperbolic tangent and linear functions are some widely used transfer function in ANN modeling.

Logistic sigmoid function: This function is a continuous function that reduces the output into the range of 0–1 and is defined as [32]:

$$\varphi(v) = \frac{1}{1 + \exp(-v)} \tag{6}$$

Hyperbolic tangent function: It is used when the desired range of output of a neuron is between -1 and 1 and is expressed as [32]:

$$\varphi(v) = \tanh(v) = \frac{1 - e^{-2v}}{1 + e^{-2v}} \tag{7}$$

Linear function: It calculates the neuron's output by simply returning the value passed to it. It can be expressed as:

$$\varphi(v) = v \tag{8}$$

4.6 Neural network architectures

The manner in which the neurons of a neural network are structured is intimately linked with the learning algorithm used to train the network. This leads to the formation of network architectures. The neural network architectures are classified into distinct classes depending upon the information flow. The different network architectures are: (a) multilayer perceptions, (b) recurrent, (c) RBF, (d) Kohonen self-organizing feature map, etc.

4.7 Multilayer perceptions (MLPs)

MLPs are layered (single-layered or multi-layered) feed forward networks typically trained with static back-propagation (**Figure 3**). Therefore, it is also called as FFBP neural networks. These networks have found their way into countless applications requiring static pattern classification. This architecture consists of input layers, output layer(s) and one or more hidden layers. The input signal moves in only forward direction from the input nodes to the output nodes through the hidden nodes. The function of hidden layer is to perform intermediate computations in between input and output layers through weights. The major advantage of FFBP is that they are easy to handle and can easily approximate any input-output map [37].

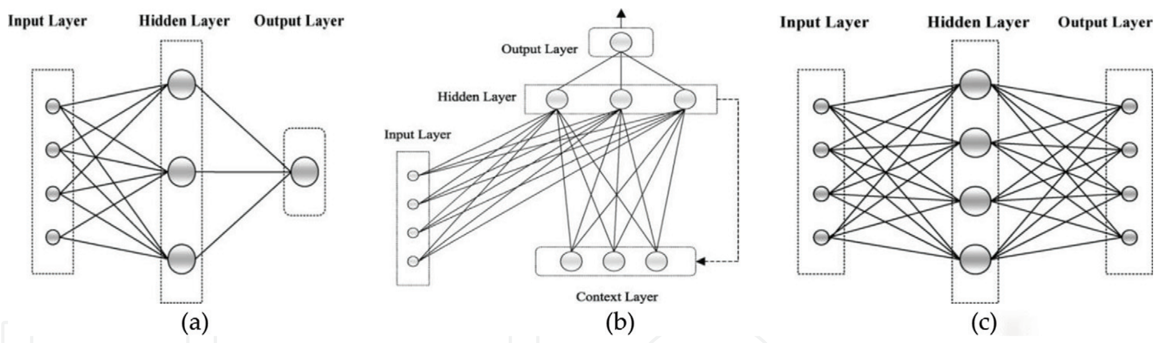

Figure 3.
Types of neural network architectures [37]. (a) Multilayer perception; (b) recurrent neural network; (c) radial basis function network.

4.8 Recurrent neural networks (RNN)

RNN may be fully recurrent networks (FRN) or partially recurrent networks (PRN). FNN sent the outputs of the hidden layer back to itself, whereas PRN initiates the fully RNN and add a feed-forward connection (**Figure 3**). A simple RNN could be constructed by a modification of the multilayered feed-forward network with the addition of a 'context layer'. At first epoch, the new inputs are sent to the RNN and previous contents from the hidden layer are passed to context layer and at next epoch, the information is fed back to the hidden layer. Similarly, weights are calculated hidden to context and vice versa. The RNN can have an infinite memory depth and thus find relationship through time as well as through the instantaneous input space. Recurrent networks are the state-of-the-art in nonlinear time series prediction, system identification, and temporal pattern classification [37–39].

4.9 Radial basis function (RBF) networks

RBF is a three-layer feed-forward network that consists of nonlinear Gaussian transfer function in between input and hidden layers and linear transfer function in between hidden and output layers (**Figure 3**). The requirement of hidden neurons for the RBF network is more as compared to standard FFBP, but these networks tend to learn much faster than MLPs [37]. The most common basis function used is Gauss function and is given by:

$$R_i = -\exp\left(-\sum_{i=1}^{n} \frac{\|x_i - c_i\|^2}{2{\sigma_{ij}}^2}\right) \tag{9}$$

where R_i = basis or Gauss function; c = cluster center; σ_{ij} = width of the Gaussian function. The centers and widths of the Gaussians are set by unsupervised learning rules, and supervised learning is applied to the output layer. After the center is determined, the connection weights between the hidden layer and output layer can be determined simply through ordinary back-propagation (gradient-descent) training. The output layer performs a simple weighted sum with a linear output and the weights of the hidden layer basis units (input to hidden layer) are set using some clustering techniques.

$$y = \sum_{i=1}^{n} w_i R_i(x_i) + w_o \tag{10}$$

where w_i = connection weight between the hidden neuron and output neuron; w_o = bias; $\mathrm{x_i}$ = input vector.

4.10 ANN learning paradigms

Broadly speaking, there are two types of learning process namely, supervised and unsupervised. In supervised learning, the network is presented with examples of known input-output data pairs, after which it starts to mimic the presented input output behavior or pattern. In unsupervised learning, the network learns on their own, in a kind of self-study without teacher.

Supervised learning: It is also called 'associative learning' involves a mechanism of providing the network with a set of inputs and desired outputs. It is like learning with the help of a teacher. The so-called teacher has the knowledge of the environment and the knowledge is represented by a set of input-output examples. The environment is, however, unknown to the neural network. The network parameters (i.e., synaptic weights and error) are adjusted iteratively in a step-by-step fashion under the combined influence of the training vector and the error signal. After the completion of training, the neural network is able to deal with the environment completely by itself [32]. In supervised learning, FFBP NN is the most popular ones. In the FFBP NNs, neurons are organized into layers where information is passed from the input layer to the final output layer in a unidirectional manner. Any network in ANN consists of 'neurons or nodes or parallel processing elements' which interconnects the each layer with weights (W). A three layer (input (i), hidden (j) and target/output (k)) FFBP NN with weights W_{ij} and W_{jk} is shown in **Figure 4**. During training the FFBP NN, the initial or randomized weight values are corrected or adjusted as per calculated error in between output and target values and back-propagates these errors (from right to left in **Figure 4**) un till minimum error criteria achieved.

Unsupervised learning: Network is provided with inputs but not with desired outputs. The system itself must then decide what features it will use to group the input data. This is often referred to as self-organization or adaption. Provision is made for a task-independent measure of the quality of representation that the network is required to learn and the free parameters of the network are optimized with respect to that measure [32]. The most widely used unsupervised neural network is the Kohonen self-organizing map, KSOM.

4.11 Kohonen self-organizing map (KSOM)

KSOM maps the input data into two-dimensional discrete output map by clustering similar patterns. It consists of two interconnected layers namely, multidimensional input layer and competitive output layer with 'w' neurons (**Figure 5**).

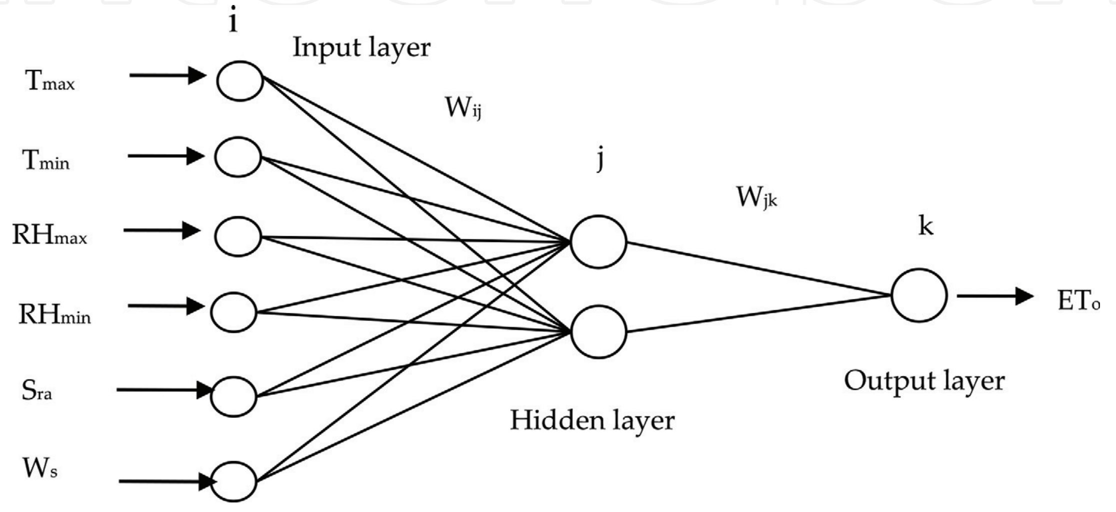

Figure 4.
A three layer feed-forward ANN model [7].

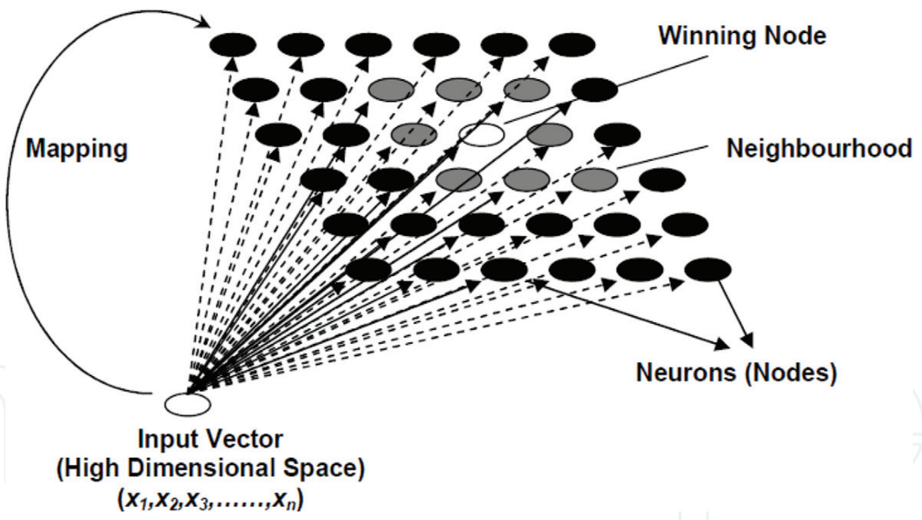

Figure 5.
Kohonen self organizing map [40].

Each node or neuron 'i' (i = 1, 2, ... w) is represented by an n-dimensional weight or reference vector $w_i = [w_{i1},\ldots,w_{in}]$. The 'w' nodes can be ordered so that similar neurons are located together and dissimilar neurons are remotely located on the map. The topology of network is indicated by the number of output neurons and their interconnections. The general network topology of KSOM is either a rectangular or a hexagonal grid. The number of neurons (map size), w, may vary from a few dozen up to several thousands, which affects accuracy and generalization capability of the KSOM. The optimum number of neurons (w) can be determined by below equation [41].

$$w = 5\sqrt{N} \tag{11}$$

where N = total number of data samples or records. Once 'w' is known, the number of rows and columns in the KSOM can be determined as:

$$\frac{l_1}{l_2} = \sqrt{\frac{e_1}{e_2}} \tag{12}$$

where l_1 and l_2 = number of rows and columns, respectively; e_1 = biggest eigen value of the training data set; e_2 = second biggest eigen value.

4.12 Training the KSOM

The KSOM is trained iteratively: initially the weights are randomly assigned. When the n-dimensional input vector x is sent through the network, the Euclidean distance between weight 'w' neurons of SOM and the input is computed by,

$$|x\text{-}w| = \sqrt{\sum_{i=1}^{n} (x_i\text{-}w_i)^2} \tag{13}$$

where x_i = ith data sample or vector; w_i = prototype vector for x_i;|denotes Euclidian distance.

The best matching unit (BMU) is also called as 'winning neuron' is the weight that closely matching to the input. The learning process takes place in between BMU and its neighboring neurons at each training iteration 't' with an aim to reduce the distance between weights and input.

$$w(t+1) = w(t) + \alpha(t)h_{lm}(x\text{-}w(t)) \tag{14}$$

where α = learning rate; *l* and m = positions of the winning neuron and its neighboring output nodes; h_{lm} = neighborhood function of the BMU *l* at iteration *t*.

The most commonly used neighborhood function is the Gaussian which is expressed as:

$$h_{lm} = \exp\left(-\frac{|l\text{-}m^2|}{2\sigma(t)^2}\right) \tag{15}$$

where *l-m* = distance between neurons *l* and *m* on the map grid; σ = width of the topological neighborhood.

The training steps are repeated until convergence. After the KSOM network is constructed, the homogeneous regions, that is, clusters are defined on the map. The KSOM trained network performance is evaluated using two errors namely, total topographic error (t_e) and quantization error (q_e).

The topographic error, t_e, is an indication of the degree of preservation of the topology of the data when fitting the map to the original data set.

$$t_e = \frac{1}{N}\sum_{i=1}^{N} u(x_i) \tag{16}$$

where $u(x_i)$ = binary integer such that it is equal to 1 if the first and second best matching units of the map are not adjacent units; otherwise it is zero.

The quantization error, q_e, is an indication of the average distance between each data vector and its BMU at convergence, that is, the quality of the map fitting to the data.

$$q_e = \frac{1}{N}\sum_{i=1}^{N} |x_i\text{-}w_{li}| \tag{17}$$

where w_{li} = prototype vector of the best matching unit for x_i.

4.13 Type of ANN training algorithms

Training basically involves feeding training samples as input vectors through a neural network, calculating the error of the output layer, and then adjusting the weights of the network to minimize the error. There are different methods for adjusting the weights. These methods are called as "training algorithms". The objective of the training algorithm is to minimize the difference between the predicted output values and the measured output values [6]. Different training algorithms are: (i) gradient descent with momentum backpropagation (GDM) algorithm, (ii) Levenberg-Marquardt (LM) algorithm, (iii) Broyden-Fletcher-Goldfarb-Shanno (BFGS) quasi Newton algorithm, (iii) resilient back propagation (RBP) algorithm, (iv) conjugate gradient algorithm, (v) one-step secant (OSS) algorithm, (vi) cascade correlation (CC) algorithm, and (vii) Bayesian regularization (BR) algorithm. The training algorithms used in this study are only briefly described below.

4.14 Gradient descent with momentum back propagation (GDM) algorithm

This method uses back-propagation to calculate derivatives of performance cost function with respect to the weight and bias variables of the network. Each variable is adjusted according to the gradient descent with momentum. The equation used for update of weight and bias is given by:

$$\Delta w_{ji}(n) = \alpha.\Delta w_{ji}(n-1) + \eta \frac{\partial E}{\partial w_{ji}} \quad (18)$$

where $\Delta w_{ji}(n)$ = correction applied to the synaptic weight connecting neuron i to neuron j; α = momentum; η = learning-rate parameter; E = error function. The equation is known as the generalized delta rule and this is probably the simplest and most common way to train a network [37].

4.15 Levenberg-Marquardt (LM) algorithm

This method is a modification of the classic Newton algorithm for finding an optimum solution to a minimization problem. In particular the LM utilizes the so called Gauss-Newton approximation that keeps the Jacobian matrix and discards second order derivatives of the error. The LM algorithm interpolates between the Gauss-Newton algorithm and the method of gradient descent. To update weights, the LM algorithm uses an approximation of the Hessian matrix.

$$W_{k+1} = W_k - \left[J^T J + \lambda I\right]^{-1} J^T e \quad (19)$$

where W = weight; e = errors; I = identity matrix; λ = learning parameter; J = Jacobian matrix (first derivatives of errors with respect to the weights and biases); J^T = transpose of J; J^TJ = Hessian matrix. For λ = 0 the algorithm becomes Gauss-Newton method. For very large λ the LM algorithm becomes steepest decent algorithm. The 'λ' parameter governs the step size and is automatically adjusted (based on the direction of the error) at each iteration in order to secure convergence. If the error decreases between weight updates, then the 'λ' parameter is decreased by a factor of λ^-. Conversely, if the error increases then 'λ' parameter is increased by a factor of λ^+. The λ^- and λ^+ are defined by user. In LM algorithm training process converges quickly as the solution is approached, because Hessian does not vanish at the solution. LM algorithm has great computational and memory requirements and hence it can only be used in small networks. It is often characterized as more stable and efficient. It is faster and less easily trapped in local minima than other optimization algorithms [37].

4.16 Online and batch modes of training

On-Line learning updates the weights after the presentation of each exemplar. In contrast, Batch learning updates the weights after the presentation of the entire training set. When the training datasets are highly redundant, the online mode is able to take the advantage of this redundancy and provides effective solutions to large and difficult problems. On the other hand, the batch mode of training provides an accurate estimate of gradient vector; convergence of local minimum is thereby guaranteed under simple conditions [23].

4.17 Multiple linear regression (MLR)

MLR technique attempts to model the relationship between two or more explanatory (independent) variables and a response (dependent) variable by fitting a linear equation to the observed data. The general form of a MLR model is given as [42]:

$$Y_i = \beta_0 + \beta_1 X_{1,\,i} + \beta_2 X_{2,\,i} + \ldots + \beta_k X_{k,\,i} + \varepsilon_i \qquad (20)$$

where Y_i = ith observations of each of the dependent variable Y; $X_{1,\,i}$, $X_{2,i}$, ⋯, $X_{k,\,i}$ = ith observations of each of the independent variables X_1, X_2, ⋯, X_k respectively; β_0, β_1, β_2, ⋯, β_n = fixed (but unknown) parameters; ε_i = random variable that is normally distributed.

The task of regression modeling is to estimate the unknown parameters (β_0, β_1, β_2, ⋯, β_n) of the MLR model [Eq. (20)]. Thus, the pragmatic form of the statistical regression model obtained after applying the least square method is as follows [42].

$$Y_i = b_0 + b_1 X_{1,\,i} + b_2 X_{2,\,i} + \ldots + b_k X_{k,\,i} + e_i \qquad (21)$$

where $i = 1, 2, \ldots, n$; b_0, b_1, b_2, ⋯, b_k estimates or unstandardized regression coefficients of β_0, β_1, β_2, ⋯, β_n respectively; e_i = estimated error (or residual) for the ith observation.

Therefore, estimate of

$$Y = \hat{Y} = b_0 + b_1 X_{1,\,i} + b_2 X_{2,\,i} + \ldots + b_k X_{k,\,i} \qquad (22)$$

The difference between the observed Y and the estimated $\hat{Y}$ is called the residual (or residual error).

The purpose of developing MLR models is to establish a simple equation which is easy to use and interpret. The MLR modeling is very useful, especially in case of limited field data. Moreover, it is versatile as it can accommodate any number of independent variables [43].

4.18 The FAO-56 Penman-Monteith method

The FAO-56 PM method is recommended as the standard method for estimating ET_o in case of locations where measured lysimeter data is not available. The equation for the estimation of daily ET_o can be written as [3]:

$$ET_o = \frac{0.408\Delta(R_n - G) + \gamma \frac{900}{T+273} W_s (e_s - e_a)}{\Delta + \gamma(1 + 0.34 W_s)} \qquad (23)$$

where ET_o = reference evapotranspiration calculated by FAO-56 PM method (mm day^{-1}); R_n = daily net solar radiation (MJ m^{-2} day^{-1}); γ = psychrometric constant (kPa °C^{-1}); Δ = slope of saturation vapor pressure versus air temperature curve (kPa °C^{-1}); e_s and e_a = saturation and actual vapor pressure (kPa), respectively; T = average daily air temperature (°C); G = soil heat flux (MJ m^{-2} day^{-1}); W_s = daily mean wind speed (m s^{-1}).

The ET_o values obtained from above equation are used as target data in ANN due to unavailability of lysimeter measured values.

5. Methodology

For the purpose of this study, 15 different climatic locations distributed over four agro-ecological regions (AERs) are selected. The selected locations are Parbhani, Kovilpatti, Bangalore, Solapur, Udaipur (semi-arid); Anantapur and Hissar (arid); Raipur, Faizabad, Ludhiana, and Ranichauri, (sub-humid); and Palampur, Jorhat, Mohanpur, and Dapoli (humid). Daily climate data of T_{min}, T_{max}, RH_{min}, RH_{max}, W_s, S_{ra} for the period of 5 years (January 1, 2001 to

Evaluation criteria	Formulae
Root Mean Squared Error (RMSE)	$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(T_i\text{-}O_i)^2}$
Coefficient of determination (R^2)	$R^2 = \frac{\left[\sum_{i=1}^{n}(O_i\text{-}\overline{O})(T_i\text{-}\overline{T})\right]^2}{\sum_{i=1}^{n}(O_i\text{-}\overline{O})^2\sum_{i=1}^{n}(T_i\text{-}\overline{T})^2}$
Ratio of average output and target ET_o values (R)	$R = \frac{\overline{O}}{\overline{T}}$
where T_i and O_i = target (FAO-56 PM ET_o) and output (ET_o resulted from MLR or ANN models) values at the ith step, respectively; n = number of data points, $\overline{T}$and $\overline{O}$ = average of target (FAO-56 PM ET_o) and output (ET_o from MLR or ANN models) values, respectively.	

Table 2.
Performance evaluations of ANN and MLR models.

December 31, 2005) was collected from All India Coordinated Research Project on Agrometeorology (AICRPAM), Central Research Institute for Dryland Agriculture (CRIDA), Hyderabad, Telangana, India. These data were used for the development and testing of various ANN-based ET_o models. Due to the unavailability of lysimeter measured ET_o values for these stations, it is estimated by the FAO-56 PM method, which has been adopted as a standard equation for the computation of ET_o and calibrating other Eqs. [10]. The normalization technique was applied to both the input and target data before training and testing such that all data points lies in between 0 and 1. The normalization process removes the cyclicity of the data. The following procedure was adopted for normalizing the input and output data sets. Each variable, $X_{i,}$ in the data set was normalized ($X_{i,\ norm}$) between 0 and 1 by dividing its value by the upper limit of the data set, $X_{i,\ max}$. Resulting data was then used for mapping.

$$X_{i,\ norm} = X_i / X_{i,\ max} \tag{24}$$

ANN simulated ET_o was converted back to original form by denormalization procedure. The data from 2001 to 2005 was splitted into training (70% of 2001–2004), validation (30% of 2001–2004), and testing (2005) sets. ANN models were trained with the LM algorithm consists of one hidden layer (sigmoid transfer function) and one output layer (linear transfer function). The parameters that were fixed after a number of trials include: RMSE = 0.0001, learning rate = 0.65, momentum rate = 0.5, epochs = 500, and initial weight range = −0.5 to 0.5. The developed various ANN models were compared with basic statistical MLR models. The developed ANN models were evaluated and compared based on different error functions described in **Table 2**. Training window of the model contains general information used for training the networks like, error tolerance, Levenberg parameter (lambda) and maximum cycles of simulation. For weights selection, two options are there, weights can be randomized or it can be read from an existing weight file of previous training.

6. Results and discussion

6.1 Development of ANN models for daily ET_o estimation

ANN model with six climatic variables (T_{max}, T_{min}, RH_{max}, RH_{min},W_S, and S_{ra}) were trained and tested to evaluate the feasibility of ANN models corresponding to FAO-56 PM conventional ET_o method for 15 individual locations in India. In order to highlight the necessity of using complex ANN models, it is necessary to show the results obtained using MLR models.

6.2 Training of ANN models for daily ET_o estimation

All the ANN models were trained as per the procedure mentioned in methodology and after each training run; three performance indices (RMSE, R^2, and R_{ratio}) were calculated, to find the optimum neural network. Several runs were used for determining the optimal number of hidden neurons with different architectural configurations. The optimum neural network was selected based on criteria such that the model has minimum RMSE and maximum R^2 values. Here, it is worth to mention that the R_{ratio} is used only to know whether the models overestimated or underestimated ET_o values. Training with higher number of hidden nodes might increase the performance of ANN models. But training with a several number of hidden nodes requires more computation time and cause complexity in architecture as it has to complete number of epochs [7]. Therefore, to avoid the above difficulty, the selection of an optimum node was fixed with a trial run of 1–15 hidden nodes only (i.e., not tried beyond 15 hidden nodes). **Figure 6** shows the relationship between RMSE and number of hidden nodes of ANN models for four locations (Parbhani, Hissar, Faizabad, and Dapoli) during training. These locations are chosen randomly from each agro-ecological region such that Parbhani, Hissar, Faizabad, and Dapoli represent semi-arid, arid, sub-humid, and humid climates, respectively.

For ANN models, the best network was resulted at a hidden node of $i + 1$ (where i = number of nodes in the input layer) for most of the locations. Thus, $i + 1$ hidden nodes are sufficient to model the ET_o process using the ANN models [13–16, 44–46]. **Table 3** shows the performance statistics of ANN models for 15 locations during training. The results pertaining to the optimal network structure of ANN models, resulted at $i + 1$ hidden nodes, are only summarized in **Table 3** for 15 locations.

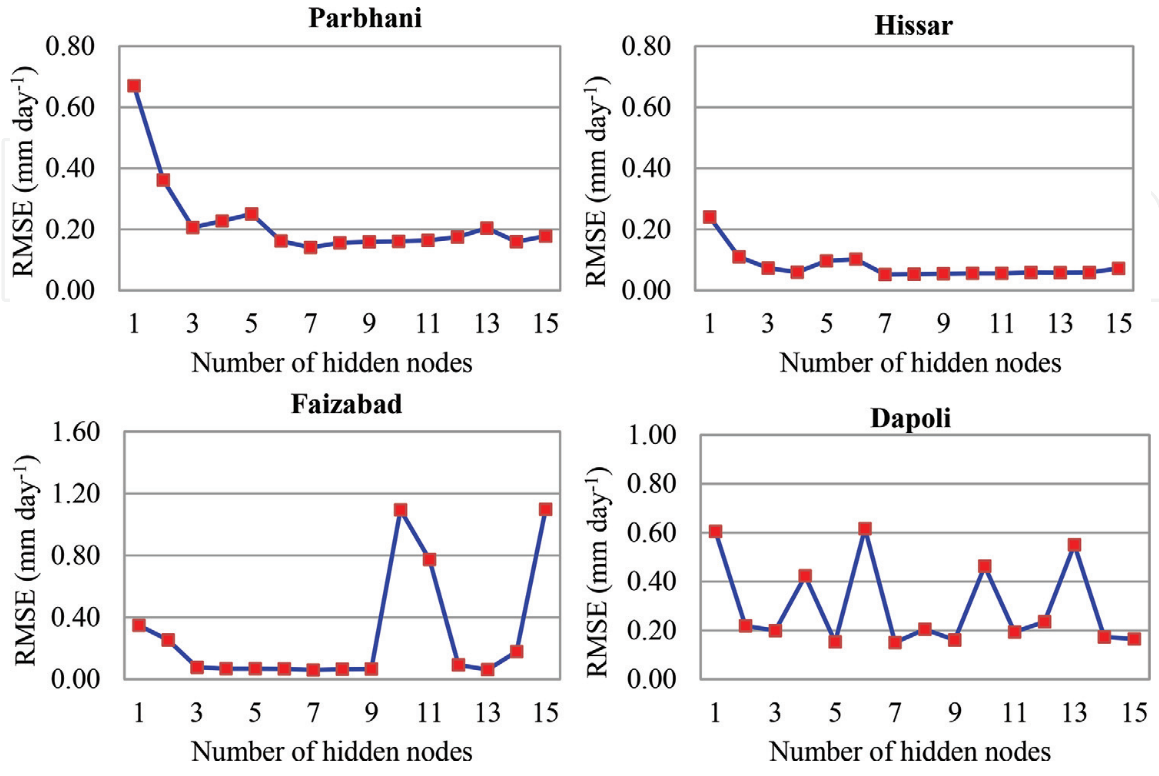

Figure 6.
RMSE variations with number of hidden nodes for ANN models.

AER	Location	ANN		
		RMSE	R^2	R_{ratio}
Semi-arid	Parbhani	0.141	0.991	0.997
	Solapur	0.271	0.969	1.000
	Bangalore	0.296	0.972	1.005
	Kovilpatti	0.254	0.991	1.000
	Udaipur	0.391	0.952	1.003
Arid	Anantapur	0.363	0.972	0.986
	Hissar	0.052	0.999	1.000
Sub-humid	Raipur	0.255	0.981	0.982
	Faizabad	0.060	0.999	1.001
	Ludhiana	0.289	0.977	0.999
	Ranichauri	0.909	0.411	1.004
Humid	Palampur	0.177	0.988	0.999
	Jorhat	0.615	0.943	1.001
	Mohanpur	0.377	0.904	1.002
	Dapoli	0.150	0.990	1.000

RMSE = mm day^{-1}; R^2 and R_{ratio} = dimensionless.

Table 3.
Performance of ANN based ET_o models during training.

AER	Location	MLR			ANN		
		RMSE	R^2	R_{ratio}	RMSE	R^2	R_{ratio}
Semi-arid	Parbhani	0.308	0.963	1.002	0.115	0.995	0.994
	Solapur	0.313	0.959	1.003	0.228	0.979	0.988
	Bangalore	0.159	0.980	1.000	0.201	0.968	0.994
	Kovilpatti	0.233	0.977	0.999	0.200	0.984	1.004
	Udaipur	0.295	0.975	1.001	0.119	0.996	0.992
Arid	Anantapur	0.275	0.977	1.000	0.222	0.984	0.998
	Hissar	0.434	0.951	0.999	0.280	0.980	1.000
Sub-humid	Raipur	0.420	0.943	1.002	0.296	0.972	1.005
	Faizabad	0.357	0.957	1.002	0.286	0.973	1.011
	Ludhiana	0.348	0.971	0.999	0.279	0.981	1.000
	Ranichauri	0.265	0.961	0.999	0.137	0.989	1.005
Humid	Palampur	0.313	0.952	1.003	0.228	0.979	1.031
	Jorhat	0.151	0.978	1.000	0.137	0.985	1.019
	Mohanpur	0.170	0.983	1.001	0.123	0.991	1.007
	Dapoli	0.177	0.973	1.001	0.152	0.981	1.009

RMSE = mm day^{-1}; R^2 and R_{ratio} = dimensionless.

Table 4.
Performance of ANN and MLR based ET_o models during testing.

6.3 FAO-56 PM-based ANN models

ET_o process is a function of various climatic factors (T_{max}, T_{min}, RH_{max}, RH_{min}, W_S, and S_{ra}). Therefore, it is pertinent to take into account the combined influence of all the climatic parameters on ET_o estimation. The ANN models corresponding to

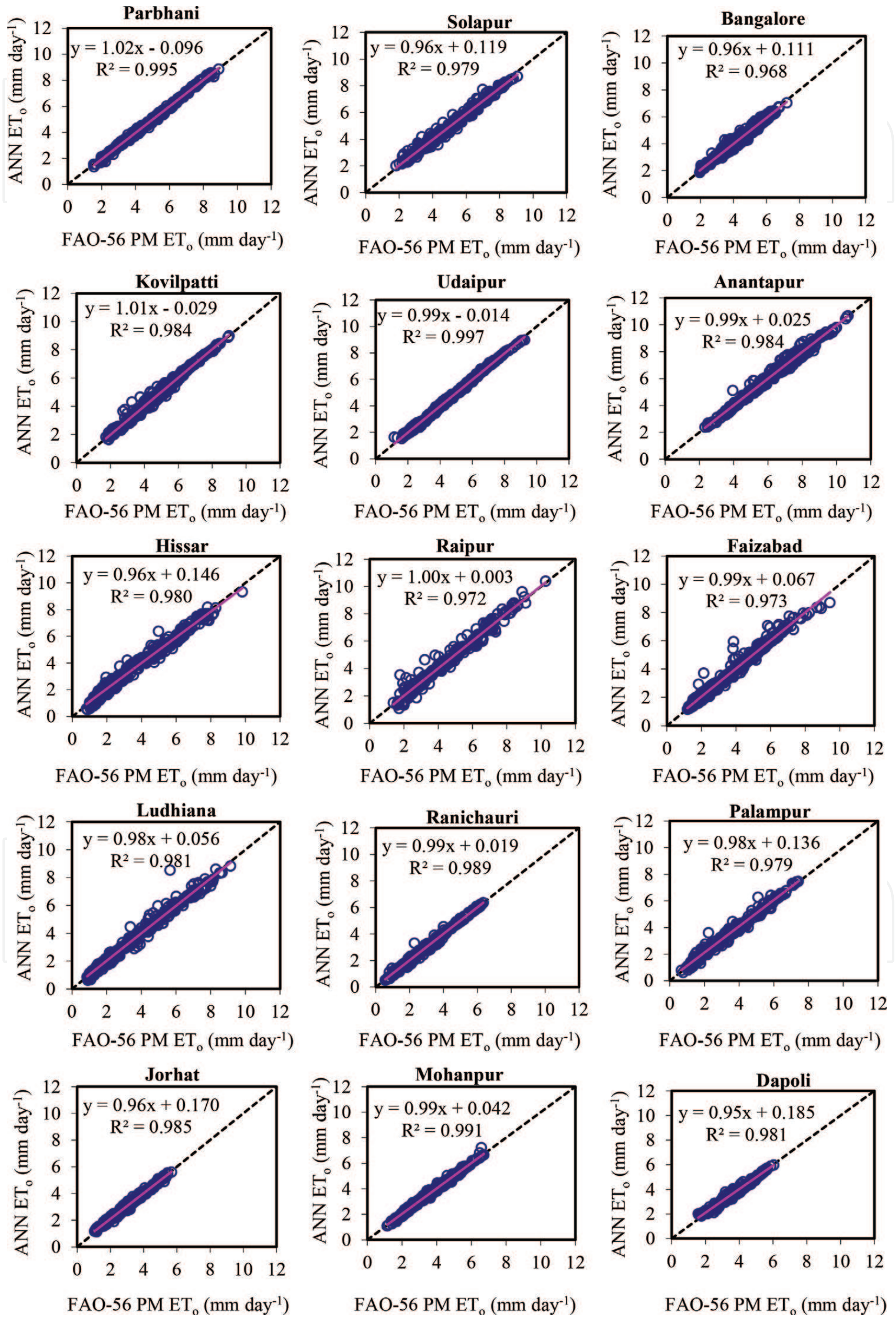

Figure 7.
Scatter plots of ANN models estimated ET_o with respect to FAO-56 PM ET_o for 15 climatic locations in India.

the FAO-56 PM were developed considering T_{max}, T_{min}, RH_{max}, RH_{min}, W_s, and S_{ra} as input and the FAO-56 PM ET_o as target. **Table 4** shows the performance statistics of ANN and MLR models for 15 locations during testing. Comparison of results obtained using MLR and ANN models indicated that the ANN models performed better than the MLR models for all locations except for Bangalore. This is confirmed from the low values of RMSE (mm day^{-1}) and high values of R^2 for ANN models as compared to the MLR models.

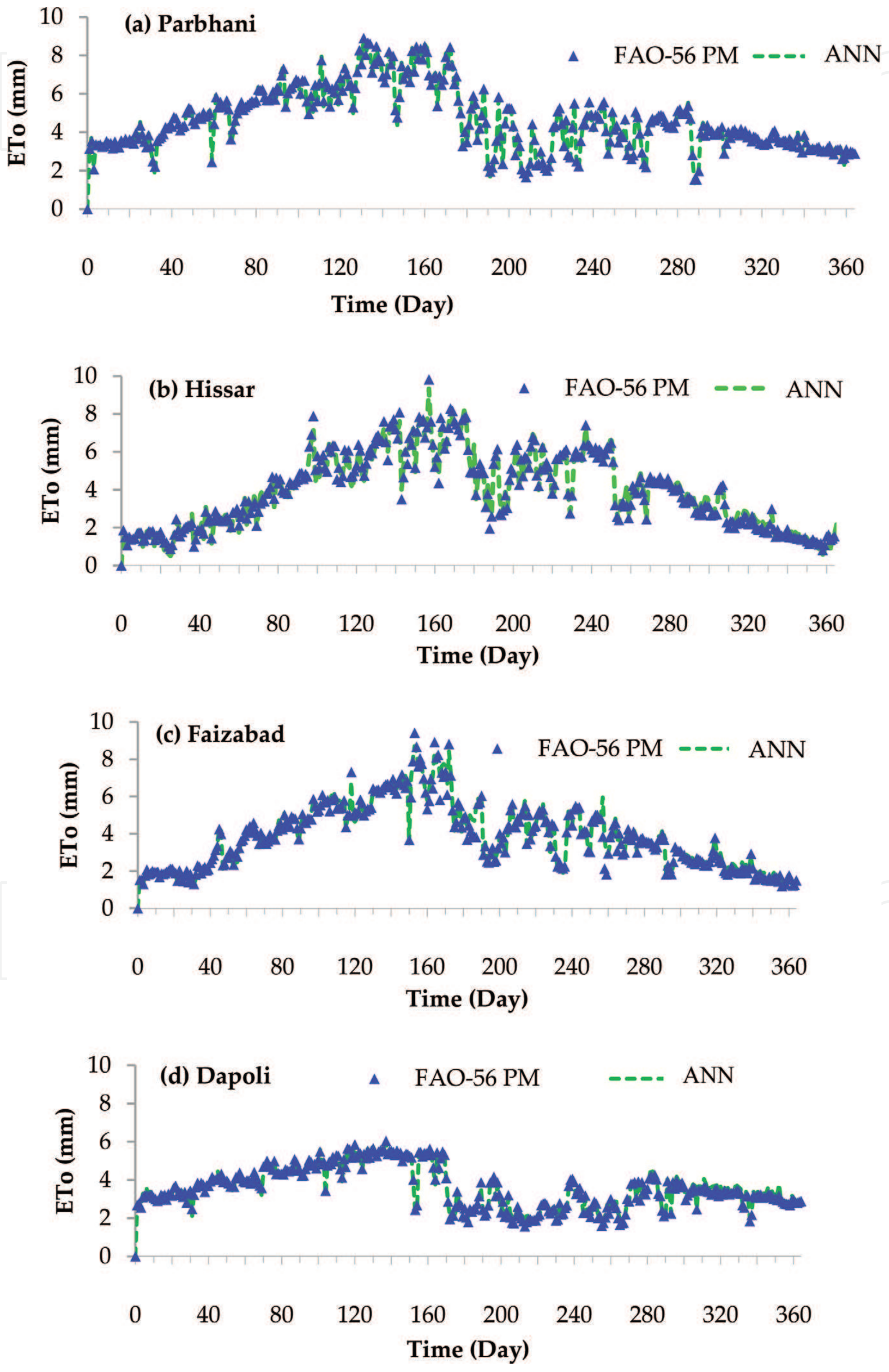

Figure 8.
Time series plots of ANN and FAO-56 PM ET_o for (a) Parbhani, (b) Hissar, (c) Faizabad, and (d) Dapoli locations.

The R_{ratio} values of MLR models for 15 locations are nearly approaching one, which simply indicates that on an average these models neither over- nor under-estimated ET_o. However, high values of RMSE and R^2 indicate that on a daily basis, these models over- and under-estimated ET_o values. Though the performance of ANN models was good as compared to MLR models, in some locations these models over- or under-estimated the ET_o values. The ANN models overestimated ($R_{ratio} > 1$) ET_o values at Palampur. The over- and under-estimations by all ANN models for the above locations were less than 3% which is negligible. The overall performance of all the models was represented as ANN > MLR for most of the locations except for Bangalore where, the performance of models was represented as MLR > ANN. The results suggest that the non-linearity of ET_o process can be adequately modeled using ANN models.

The scatter plots of the FAO-56 PM ET_o and ET_o estimated with the ANN models for 15 climatic locations in India are shown in **Figure 7**. The scatter plots confirm the statistics given in **Table 4**. Regression analysis was performed between the FAO-56 PM ET_o and ET_o estimated with the ANN and the best-fit lines are shown in **Figure 7**. The values of R^2 for ANN models were found to be >0.968. The fit line equations ($y = a_0x + a_1$) in **Figure 7** gave the values of a_0 and a_1 coefficients closer to one and zero, respectively. Due to the superior performance of ANN models over the MLR models, the time series plots of these models with 1 year data (during testing) for four selected locations Parbhani, Hissar, Faizabad, and Dapoli are shown in **Figure 8**. The location figures indicated that, ET_o estimated using ANN models matched well with the FAO-56 PM ET_o except for a few peak values in case of Faizabad.

7. Summary and conclusions

Evapotranspiration is an important and one of the most difficult components of the hydrologic cycle to quantify accurately. Prior to designing any irrigation system, the information on crop water requirements or crop evapotranspiration is needed, which can be calculated using reference evapotranspiration. There exist direct measurement methods (lysimeters) and indirect estimation procedures (physical and empirical based) for modeling ET_o. Direct methods have the limitations of arduous, cost-effective, and lack of skilled manpower to collect accurate measurements. The difficulty in estimating ET_o with the indirect physically based methods is due to the limitations of unavailability of all necessary climate data, whereas the application of empirical methods are limited due to unsuitability of these methods for all climatic conditions and need of local calibration. ANNs are efficient in modeling complex processes without formulating any mathematical relationships related to the physical process. This study was undertaken to develop ANN models corresponding to FAO-56 PM conventional ET_o method for 15 individual stations in India.

The potential of ANN models corresponding to the FAO-56 PM method was evaluated for 15 locations. The ANN models were developed considering six inputs (T_{max}, T_{min}, RH_{max}, RH_{min}, W_s, and S_{ra}) and the FAO-56 PM ET_o as target. The optimum number of hidden neurons was finalized with a trial of 1–15 hidden nodes. The ANN models gave lower RMSE values at $i + 1$ (i = number of inputs) hidden nodes for estimating ET_o. Comparison results of MLR and ANN models indicated that the ANN models performed better for all locations. However, on an average the over- and under-estimations of ET_o (<3% which is negligible) estimated by using MLR models was less as compared to ANN models. In brief, based on the above discussion on ET_o modeling, the following specific conclusions are drawn:

- For estimating ET_o using ANN model, a network of single hidden layer with $i + 1$ (i = number of input nodes) number of hidden nodes was found as adequate.
- ANN-based ET_o estimation models performed better than the MLR models for all locations.

However, it should be noted that only climate data from different agro-ecological regions of India was used in this analysis and the results might be different for various climates in other countries.

Author details

Sirisha Adamala
Natural Resources Management Division, Indian Council of Agricultural Research (ICAR) – Central Island Agricultural Research Institute (CIARI), Port Blair, Andaman & Nicobar Islands, India

*Address all correspondence to: sirisha.cae@gmail.com

References

[1] Dingman SL. Physical Hydrology. Upper Saddle River, NJ: Prentice-Hall, Inc.; 2002. p. 401

[2] Smith DM, Jarvis PG, Odongo JCW. Energy budgets of windbreak canopies in the Sahel. Agricultural and Forest Meteorology. 1997;**86**:33-49

[3] Allen RG, Pereira LS, Raes D, Smith M. Crop evapotranspiration: Guidelines for computing crop water requirements. Irrigation and Drainage Paper No. 56. Rome: FAO, 1998

[4] Debnath S, Adamala S, Raghuwanshi NS. Sensitivity analysis of FAO-56 Penman-Monteith method for different agro-ecological regions of India. Environmental Processes. 2015;**2**(4): 689-704

[5] Izadifar Z, Elshorbagy A. Prediction of hourly actual evapotranspiration using neural networks, genetic programming, and statistical models. Hydrological Processes. 2010. DOI: 10.1002/hyp.7771

[6] Adamala S, Raghuwanshi NS, Mishra A, Tiwari MK. Development of generalized higher-order synaptic neural based ET_o models for different agroecological regions in India. Journal of Irrigation and Drainage Engineering. 2014. DOI: 10.1061/(ASCE) IR.1943-4774.0000784

[7] Adamala S, Raghuwanshi NS, Mishra A, Tiwari MK. Evapotranspiration modeling using second-order neural networks. Journal of Hydrologic Engineering. 2014;**19**(6):1131-1140

[8] Adamala S, Raghuwanshi NS, Mishra A. Generalized quadratic synaptic neural networks for ET_o modeling. Environmental Processes. 2015;**2**(2): 309-329

[9] Adamala S, Raghuwanshi NS, Mishra A, Tiwari MK. Closure to evapotranspiration modeling using second-order neural networks. Journal of Hydrologic Engineering. 2015;**20**(9): 07015015

[10] Adamala S, Raghuwanshi NS, Mishra A, Singh R. Generalized wavelet-neural networks for evapotranspiration modeling in India. ISH Journal of Hydraulic Engineering. 2017:1-13

[11] Adamala S. Temperature based generalized wavelet-neural network models to estimate evapotranspiration in India. Information Processing in Agriculture. 2017;**5**(1):149-155

[12] Adamala S, Srivastava A. Comparative evaluation of daily evapotranspiration using artificial neural network and variable infiltration capacity models. Agricultural Engineering International: CIGR Journal. 2018;**20**(1):32-39

[13] Chauhan S, Shrivastava RK. Performance evaluation of reference evapotranspiration estimation using climate based methods and artificial neural networks. Water Resources Management. 2009;**23**:825-837

[14] Kumar M, Bandyopadhyay A, Raghuwanshi NS, Singh R. Comparative study of conventional and artificial neural network-based ET_o estimation models. Irrigation Science. 2008;**26**: 531-545

[15] Kumar M, Raghuwanshi NS, Singh R. Development and validation of GANN model for evapotranspiration estimation. Journal of Hydrologic Engineering. 2009;**14**:131-140

[16] Kumar M, Raghuwanshi NS, Singh R, Wallender WW, Pruitt WO. Estimating evapotranspiration using artificial neural network. Journal of Irrigation and Drainage Engineering. 2002;**128**(4):224-233

[17] Landeras G, Ortiz-Barrredo A, Lopez JJ. Comparison of artificial neural network models and empirical and semi-empirical equations for daily reference evapotranspiration estimation in Barque country (Northen Spain). Agricultural Water Management. 2008; **95**:553-565

[18] Rahimikhoob A. Artificial neural network estimation of reference evapotranspiration from pan evaporation in a semi-arid environment. Irrigation Science. 2008;**27**(1):35-39

[19] Rahimikhoob A. Comparative study of Hargreaves's and artificial neural network's methodologies in estimating reference evapotranspiration in a semiarid environment. Irrigation Science. 2008;**26**(3):253-259

[20] Rahimikhoob A. Estimating daily pan evaporation using artificial neural network in a semi-arid environment. Theoretical and Applied Climatology. 2009;**98**:101-105

[21] Rahimikhoob A. Estimation of evapotranspiration based on only air temperature data using artificial neural networks for a subtropical climate in Iran. Theoretical and Applied Climatology. 2010;**101**:83-91

[22] Sudheer KP, Gosain AK, Ramasastri KS. Estimating actual evapotranspiration from limited climatic data using neural computing technique. Journal of Irrigation and Drainage Engineering. 2003;**129**(3):214-218

[23] Sudheer KP, Gosain AK, Rangan DM, Saheb SM. Modeling evaporation using an artificial neural network algorithm. Hydrological Processes. 2002;**16**:3189-3202

[24] Trajkovic S. Temperature based approaches for estimating reference evapotranspiration. Journal of Irrigation and Drainage Engineering. 2005;**131**(4): 316-323

[25] Zanetti SS, Sousa EF, Oliveira VPS, Almeida FT, Bernardo S. Estimating evapotranspiration using artificial neural network and minimum climatological data. Journal of Irrigation and Drainage Engineering. 2007;**133**: 83-89

[26] Han H, Felker P. Estimation of daily soil water evaporation using an artificial neural network. Journal of Arid Environments. 1997;**37**:251-260

[27] Bruton JM, Mcclendon RW, Hoogenboom G. Estimating daily pan evaporation with artificial neural networks. Transactions of ASAE. 2000; **43**(2):491-496

[28] Trajkovic S, Stankovic M, rB T. Estimation of FAO Blaney-Criddle b factor by RBF network. Journal of Irrigation and Drainage Engineering. 2000;**126**(4):268-270

[29] Trajkovic S, Todorovic B, Stankovic M. Forecasting reference evapotranspiration by artificial neural networks. Journal of Irrigation and Drainage Engineering. 2003;**129**(6): 454-457

[30] Michael AM. Irrigation: Theory and Practice. New Delhi: Vikas Publishing House; 2011. p. 768

[31] McCulloch WS, Pitts WA. Logical calculus of the ideas imminent in nervous activity. Bulletin of Mathematical Biophysics. 1943;**5**: 115-133

[32] Haykin S. Neural Networks-a Comprehensive Foundation. 2nd ed. Upper Saddle River, NJ: Prentice-Hall; 1998. p. 205

[33] Caudill M. Neural Networks Primer I. AI Expert; 1987

[34] Caudill M. Neural networks primer II, III, IV and V. 1988; AI Expert

[35] Caudill M. Neural Networks Primer VI, VII and VIII. AI Expert; 1989

[36] Hecht-Nielsen R. Neurocomputing. Addison-Wesley; 1990 ISBN 0-201-09255-3

[37] Daliakopoulos IN, Coulibaly P, Tsanis IK. Groundwater level forecasting using artificial neural network. Journal of Hydrology. 2005; **309**:229-240

[38] ASCE Task Committee on Artificial Neural Networks in Hydrology. Artificial neural networks in hydrology I: Preliminary concepts. Journal of Hydrologic Engineering. 2000a;**5**(2): 115-123

[39] ASCE Task Committee on Artificial Neural Networks in Hydrology. Artificial neural networks in hydrology II: Hydrologic applications. Journal of Hydrologic Engineering. 2000b;**5**(2): 124-137

[40] Adeloye AJ, Rustum R, Kariyama ID. Neural computing modeling of the reference crop evapotranspiration. Environmental Modelling and Software. 2012;**29**:61-73

[41] Garcia H, Gonzalez L. Self-organizing map and clustering for wastewater treatment monitoring. Engineering Applications of Artificial Intelligence. 2004;**17**(3):215-225

[42] Makridakis S, Wheelwright SC, Hyndman RJ. Forecasting Methods and Applications. 3rd ed. Singapore: John Wiley and Sons (Asia) Ltd.; 2008. p. 656

[43] Hodgson FDI. The use of multiple linear regression in simulating ground-water level responses. Ground Water. 1978;**16**(4):249-253

[44] Abedi-Koupai J, Amiri MJ, Eslamian SS. Comparison of artificial neural network and physically based models for estimating of reference evapotranspiration in greenhouse. Australian Journal of Basic and Applied Sciences. 2009;**3**(3):2528-2535

[45] Kisi O. Evapotranspiration modeling from climatic data using a neural computing technique. Hydrological Processes. 2007;**21**(4): 1925-1934

[46] Kisi O. The potential of different ANN techniques in evapotranspiration modeling. Hydrological Processes. 2008; **22**:2449-2460