International Journal of Environmental Science and Technology
https://doi.org/10.1007/s13762-021-03565-y

ORIGINAL PAPER

An assessment of multi-layer perceptron networks for streamflow forecasting in large-scale interconnected hydrosystems

V. A. D. de Faria[1] · A. R. de Queiroz[2] · L. M. Lima[3] · J. W. M. Lima[4] · B. C. da Silva[5]

Received: 14 February 2021 / Revised: 8 July 2021 / Accepted: 17 July 2021

Abstract

This work analyzes the use of artificial neural networks in the short-term streamflow forecasting for large interconnected hydropower systems. The state-of-the-art optimization algorithms, activation functions, and weight initialization techniques are investigated together with classic methods. We present an algorithm to define the neural network inputs in large hydrosystems and apply it to create models for 55 major hydro plants located in the Paraná Basin, which contribute to more than 30% of the total power generated in Brazil. The paper also compares the performance of the neural networks with the hydrological models that are currently used by the independent system operator to define the dispatch of the electric power generators. Our results show that, overall, the neural network models provide more accurate forecasts than the hydrological models used by the Brazilian System Operator. Finally, the paper discusses the contributions of historical rainfall information in the forecasting of streamflow while using neural network models.

Keywords Artificial neural networks · Hydropower generation · Hydrological run-off models · Large interconnected hydrosystems

Abbreviations

Adam	Adaptive moment estimation
ANN	Artificial neural network
AR	Autoregressive
ARMA	Autoregressive moving average
CPU	Central processing unit
GD	Gradient descent
GDM	Gradient descent with momentum
GPU	Graphics processing unit
ISO	Independent system operator
IVS	Input variable selection
MAPE	Mean absolute percentage error
MGB-IPH	Large-scale hydrological model
MLP	Multi-layer perceptron
MSE	Mean square error
NSE	Nash–Sutcliffe efficiency
OWI	Optimal weight initialization
PAR	Periodic autoregressive
PARMA	Periodic autoregressive moving average
ReLU	Rectified linear unit
RMSprop	Root-mean-square propagation
SMAP	Soil moisture accounting procedure
SVM	Support vector machine
VELMA	Visualizing ecosystem land management assessments
WI	Weight initialization

Indices and sets

In	Set of input candidates
$\mathrm{In}_{\mathrm{S}}^{*}$	Best set of streamflow inputs used in a specific ANN
$\mathrm{Is}_{t_{\mathrm{I}}}$	Input streamflow candidate (s_{I}, t_{I})
$\mathrm{Ip}_{t_{\mathrm{I}}}^{D_{\mathrm{AR}}}$	Input rainfall candidate accumulated through D_{AR} days (p_{I}, t_{I}); $D_{\mathrm{AR}} = 1$: nonaccumulated rainfall
$\mathrm{Os}_{t_{\mathrm{o}}}$	Output streamflow (s_{O}, t_{O})

Editorial responsibility: Maryam Shabani.

✉ V. A. D. de Faria
vaduraes@ncsu.edu

[1] Operations Research Department, North Carolina State University, Raleigh, NC, USA

[2] Department of Civil, Construction, and Environmental Engineering, North Carolina State University, Raleigh, NC, USA

[3] Duke University's Nicholas School of the Environment, Duke University, Durham, NC, USA

[4] Department of Electrical Engineering, Federal University of Itajubá, Itajubá, MG, Brazil

[5] Institute of Natural Resources, Federal University of Itajubá, Itajubá, MG, Brazil

Published online: 23 July 2021

$p_I \in \text{In\_}p$	Set of rainfall stations considered for input candidates
$s_I \in \text{In\_}s$	Set of streamflow stations considered for input candidates
s_O	Output streamflow station
t_I	Number of days preceding the first day of forecast
t_O	Day of forecast

Functions

C	Component of the cost function that penalizes the difference between $y^{(i)}$ and $\hat{y}^{(i)}$
$\text{Cr}(A, B)$	Function that computes the correlation coefficient between the variables A and B
$E_{\text{dev}}(k)$	Error in the development set until the kth epoch simulation
J	Cost function with ℓ_2 regularization
$M2$	Cost function (28)
$R2$	Cost function (29)
$U(a_1, a_2)$	Continuous uniform distribution in the interval $[a_1, a_2]$
$U_D(a_1, a_2)$	Discrete uniform distribution in the interval $[a_1, a_2]$
$\text{Var}[w^\ell]$	Variance of the weights in the ℓth layer
w_2^2	Square of the ℓ_2 norm of the weight matrixes
$\sigma(x)$	Logistic sigmoid activation function

Variables and parameters

b	Bias vector
$C_{\text{IS/OS}}$	Lowest correlation coefficient between the streamflow input candidates and output streamflow
C_{IR}	Maximum correlation coefficient for the input rainfall candidates
$C_{\text{IR/OS}}$	Lowest correlation coefficient between the rainfall input candidates and output streamflow
C_{IS}	Maximum correlation coefficient for the input streamflow candidates
db	Gradients of the bias vector
dw	Gradients of the weights
D_{AR}	Number of days lag that rainfall is accumulated
D_{IS}	Number of days lag of input streamflow investigated
D_{OS}	Number of days of output streamflow
D_{IR}	Number of days lag of input rainfall investigated
$\overline{\text{Epoch}}$	Maximum number of epochs
k_{prob}	Regularization hyperparameter in dropout
m	Number of output variables
n	Number of output examples
n_ℓ	Number of neurons in the ℓth layer
N_{Hyp}	Number of times that different hyperparameters are tested
N_{RC}	Number of times that different $C_{\text{IR/OS}}$ and C_{IR} correlation coefficients are tested
N_{SC}	Number of times that different $C_{\text{IS/OS}}$ and C_{IS} correlation coefficients are tested
$\underline{w}$	Weights
$\overline{y^{(i)}}$	Average streamflow for the ith example
$y^{(i,t)}$	Estimate for the tth output variable (day of forecast) in the ith example
$\hat{y}^{(i,t)}$	Expected value for the tth output variable (day of forecast) in the ith example
α	Learning rate variable
λ	ℓ_2 Regularization hyperparameter
μ_0	Mean of the output variables $\hat{y}$
σ_0	Standard deviation of the output variables $\hat{y}$

Introduction

The use of artificial intelligence techniques to support decision making has grown substantially and reached higher levels of quality and robustness in the last decade. This fact has been driven by the fast advance of nanotechnology followed by cost reductions and processing capacity boost of components such as GPUs and CPUs. In this context, the increase in data availability and quality allowed a wide dissemination of big data analytics techniques such as artificial neural networks (ANNs) (Russom 2011; Najafabadi et al. 2015) in several areas of engineering. Applications of ANNs can be found in load and wind power forecasting in (Quan et al. 2014), commodity price forecasting (Li et al. 2018), and many others (Tian et al. 2018; Jo et al. 2015). The enormous potential in this field of study has made companies such as Google provides open-source frameworks such as TensorFlow (TensorFlow 2019), specialized in the development of ANN models with a strong support for deep learning, which contributed even more to a faster development of this research field. Compared to other shallow ANN models, deep learning was proven to be consistently superior while dealing with complex problems such as image classification (Krizhevsky et al. 2012) and speech recognition (Hinton et al. 2012a).

The first applications of ANNs in streamflow forecasting are related to the end of the twentieth century (Zealand et al. 1999; Hsu et al. 1995). By that time, researchers had already noticed the potential of applying ANNs in hydrology. However, limitations in computer performance and data availability prevented a more effective adoption of these studies in practical applications for supporting planning and operations decisions associated with hydropower generation. Nowadays, several of the previous limitations were completely eliminated or severely reduced. The advances

in nanotechnology allowed the development of CPUs and GPUs several times faster and significantly more accessible in terms of costs. This new environment generated the conditions for an efficient use of ANNs techniques in real applications. While the widespread of the internet and cheap storage systems contributed to the advances in big data analytics, it also played a fundamental role with respect to the availability of climate and hydrological data which can be better used to enhance data-driven streamflow forecasting models.

Water resources are of great importance for the electricity supply in countries and regions where the availability of hydropower is relevant (Faria et al. 2018). For example, in Brazil hydroelectricity corresponds to near 70% of the total generated energy and hydropower installed capacity is approximately 100 GW (ONS 2017) (65% of the total generation capacity), characterizing a hydrodominated power system (de Queiroz et al. 2016). In China, hydropower only accounts for about 20% of the total generated energy; however, this country has the highest installed hydropower capacity in the world (341 GW), making water a strategic resource (Iha 2018). Globally, hydropower is the world's largest source of renewable generation, accounting for more than 60% of the total renewable energy generated in 2017 (REN21 2018). This high dependence on hydropower resources raises the need for developing better models to forecast streamflows that are influenced by hydrological trends and future weather conditions.

The models used in streamflow forecasting can be divided into three main categories, namely empirical, conceptual, and physical based (Eslamian 2014). In empirical models, the relationship between input and output variables is established statistically, without any reference to the laws of the hydrological process. Commonly used empirical models are ANNs (Tongal and Booij 2018; Bravo et al. 2009), support vector machines (SVM) (Tongal and Booij 2018; Yaseen et al. 2016), and some linear time series models (Box et al. 2015) such as autoregressive (AR), AR moving average (ARMA), periodic AR (PAR), periodic ARMA (PARMA) (Rasmussen et al. 1996), and the dynamic linear model (Lima et al. 2014). Conceptual models are based on the continuing equations of water and represent an intermediate modeling between the empirical and the physical models. In what concern the representation of South America, the models SMAP (Lopes et al. 1982; Lopes and Montenegro 2017) and MGB-IPH (Pontes et al. 2017) are two well-known examples of conceptual hydrological models. Finally, the physical models make use of physical laws such as conservation of mass, energy, and momentum in order to represent the hydrological processes (Sitterson et al. 2017).

From the investigated literature, empirical, conceptual, and physical-based models perform differently in terms of accuracy depending on the characteristics of the region analyzed, data availability, data quality, and timescale (Eslamian 2014). Empirical models such as ANNs and SVMs are usually less reliable than conceptual and physical-based models when forecasting scenarios too different from those used in their training set; therefore, they are not frequently used to analyze anomalous hydrological conditions. Empirical models are also very dependent on the size of their training data and tend to perform exceptionally well in large datasets and poorly in small datasets; this limitation is becoming less relevant with the increase in the availability and resolution of measured hydrological data. Finally, conceptual and physical-based models require a good representation and understanding of the basin characteristics including soil type, topology, relief, climate conditions, and others; therefore, uncertainties in this data representation can significantly affect the accuracy of these models.

In what concerns streamflow forecasting problems modeled using ANNs, a large number of papers have focused on the comparison of this technique using different data preprocessing strategies and different forecasting methods. For example, Santos and Silva 2014 apply a wavelet transform in the input streamflow and show relevant improvements of the ANN performance when compared to models that do not use this technique. The work of (Tongal and Booij 2018) investigates the performance of ANN, SVM, and random forest models, while using a base flow separation strategy. The work of (Chen et al. 2014) uses the copula-entropy theory to identify optimal inputs for ANNs, while Bravo et al. 2009 compares the performance of an ANN model with the conceptual hydrological model MBG-IPH, where ANNs showed lower forecasting skills at the time when compared to the conceptual model.

Despite the extensive literature available on ANNs applied to streamflow forecasting, few of these works have investigated the influence of different optimization models, activation functions, and weight initialization techniques in their application. These characteristics are usually selected arbitrarily, based on the few works that have evaluated them (KİŞİ 2007; Zadeh et al. 2010), or are based on general results from other ANN applications. Although this approach ends up saving modeling and simulation time, it may lead to a drastic reduction of computational efficiency and accuracy. Also, with the fast advance of deep learning, new techniques such as the ones developed in (Kingma and Ba 2015; Glorot and Bengio 2010) are frequently being proposed and showing promising benefits in different applications. Therefore, the streamflow forecasting literature using ANNs must be updated to provide reasonable insights over the benefits of incorporating these new algorithms and strategies.

This work focuses on short-term natural streamflow forecasting, in the range of two weeks ahead, using multi-layer perceptron ANNs. One of the original contributions of this piece is the extensive evaluation of new approaches that can be used to improve the quality of the information available

for hydropower operational planning. Although this paper gives greater attention to ANN models that use as input only historical streamflow, it also discuss the marginal benefit of incorporating rainfall information as input for some hydro plants. Another significant contribution of this paper is the evaluation of different optimization techniques used in the ANN training process, as well as activation functions and weight initialization (WI) methods. An algorithm to define the ANNs inputs in large hydrosystems is also applied to evaluate ANN-based forecasting models in a real study case composed of 55 hydro plants located in the *Paraná Basin* in Brazil. To our knowledge this is the first work that investigates the use of ANNs for forecasting water inflows in large-scale systems.

Materials and methods

This work employs a feedforward neural network architecture called multi-layer perceptron (MLP) that maps a set of input values to output values. This ANN is called feedforward because the information flows from the input layer to the output layer without feedback connections. In contrast, an ANN in which information from the output is fed back into the input is called recurrent neural network (Kumar et al. 2004; Sttari et al. 2012). MLPs are commonly used in several areas of forecasting and, together with the recurrent neural networks, have shown an excellent performance in numerous applications.

Figure 1 presents a schematic of an MLP. The input layer receives the ANN inputs ($i^{(k)}$) that are traditionally selected based on the modeler experience or through a statistical process. Formally, this step is called input variable selection (IVS). After passing through the input layer the information is transmitted to the hidden layers where a series of computations are performed. The neurons on the hidden layers, represented by circles, make linear combinations of the information arriving, apply a nonlinearity to the results, and transmit this information to the next layer of neurons. This nonlinearity is formally called activation function and is discussed in a different section. Finally, after leaving the hidden layers, the information passes through the output layer where its neurons apply a final linear combination to the information that arrived and generate an estimate (prediction) for the output variables. In some applications, a nonlinearity is also applied at the output layer such as in classification problems (Goodfellow et al. 2016).

Overall, the main goal in an MLP is to calibrate the parameters w (weight) and b (biases), responsible for the linear combinations, in a way that the difference between the ANN estimate and expected value is minimized.

Data segmentation and normalization

In ANN models, the data that relate input/output historical information are called learning examples. Traditionally, the set of learning examples is divided into three sets, called training (train), development (dev), and test sets. The objective of this segmentation is to provide independent data to optimize the weight and biases of the ANN model (training set); to make interactive adjustments in the ANN hyperparameters and validate the training, that is, verify whether the behavior observed in the training set can be generalized to other sets (dev set); and finally to estimate the performance of the ANN (test set). The percentage of the data allocated to each of these sets depends on the complexity of the model, and amount of data available, two common segmentation strategies for small datasets, up to 10,000 examples, is 60% train set, 20% dev set, and 20% test set (60/20/20), and (70/20/10).

Additionally, in many ANN applications, it is a common practice to shuffle the learning examples before dividing them in each set (Goodfellow et al. 2016), but this is not the case in streamflow forecasting studies (Santos and Silva 2014; Tongal and Booij 2018). Due to the temporal interaction of each learning example, the data shuffle indirectly transfers relevant information of the dev and test sets to the training set, leading to an overestimation of the ANN performance. In this case, the ANN would present an average error during forecasting (after training) much higher than its

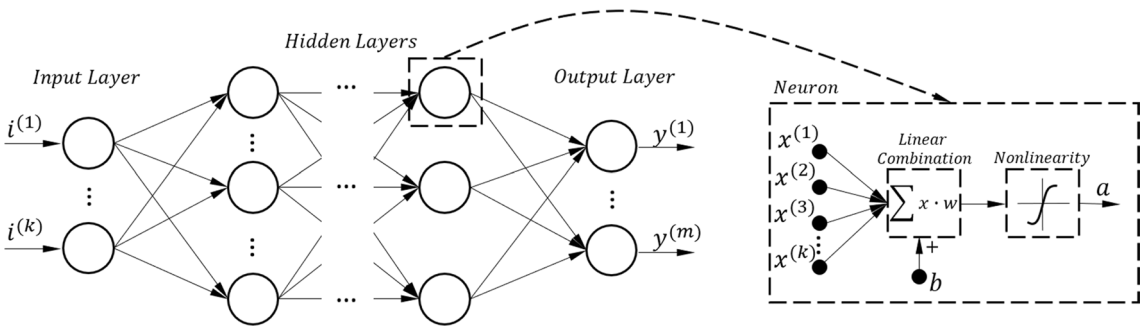

Fig. 1 Example of multi-layer perceptron

test set error. Additionally, due to changes in climate, soil, and vegetation it is interesting to choose a development set that is closer to the present so that the tuning of the ANN architecture using the dev set would also lead to an ANN model that better represents the present.

Normalization is also key in the ANN data preparation since it tends to speed up the optimization algorithms by reducing the dependence of the gradients on the scale of the parameters. A common way of normalizing data is by rescaling it to have a predefined mean and variance (standardization), typically mean 0 and variance 1 (1) (Géron 2017; Aggarwal 2018). This was the normalization adopted in this work.

$$X_{\mathrm{N}} = \frac{X - \mathrm{mean}(X)}{\mathrm{std}(X)} \tag{1}$$

Optimization algorithms

There are several optimization algorithms applied for training ANN models, but the gradient descent with momentum (GDM) (Polyak 1964), the root-mean-square propagation (RMSprop) (Hinton et al. 2012b), and the adaptive moment estimation (Adam) (Kingma and Ba 2015; Géron 2017) certainly had a significant influence in the recent explosion and success of machine learning applications. The idea behind the ANN learning process is related to the backpropagation algorithm (Géron 2017). After providing a weight initialization for the ANN parameters and completing a forward propagation step, going from the input layer to the output layer computing linear combinations, and applying the results to activation functions, all outputs/inputs of each neuron and the value of the cost function are obtained. With this information, it is possible to apply the backpropagation algorithm where the chain rule (Goodfellow et al. 2016) is used to compute the gradients of the weights and biases from the output layer to the input layer. In the gradient descent method (GD), the weight and biases are updated by making small steps in the gradient direction. After these updates, the process mentioned above repeats until a certain convergence criterion is reached. Equations (2) and (3) describe the weight and bias updates, respectively, while using the GD algorithm, α is the learning rate, and dw and db are the gradients for the weight and biases.

$$w = w - \alpha \cdot dw \tag{2}$$

$$b = b - \alpha \cdot db \tag{3}$$

The GDM, the RMSprop, and the Adam algorithms follow the same steps as mentioned for the GD except for the weight and biases update. The GDM basically computes an exponentially weighted average of the gradients and uses this value to update the weights and biases. Equations (4)–(7) describe the w and b updates. The variables V_{dw} and V_{db} are initialized with zero, and β_1 is a parameter that control the number gradients the model averages over (Géron 2017). A common value for β_1 is 0.9, what in practice means averaging over the previous ten gradients (Géron 2017; Ruder 2017). By performing the update in such a way, the GDM tends to smooth oscillations of the gradients, leading a faster learning process when compared to the traditional GD method.

$$V_{dw} = \beta_1 \cdot V_{dw} + dw \tag{4}$$

$$V_{db} = \beta_1 \cdot V_{db} + db \tag{5}$$

$$w = w - \alpha \cdot V_{dw} \tag{6}$$

$$b = b - \alpha \cdot V_{db} \tag{7}$$

The RMSprop computes an exponentially weighted average of the square of the gradients and uses this information to indirectly adjust the learning rate (Hinton et al. 2012b). Among its characteristics, the algorithm is especially useful in escaping from plateaus with small gradients (Morse and Stanley 2016) and usually presents a better performance when compared to the GDM (Géron 2017; Ruder 2017; Dozat 2016). Equations (8)–(11) describe the w and b updates for this algorithm; S_{dw} and S_{db} are initialized with zero; dw^2 and db^2 are the square of the gradient with respect to w and b, respectively, an elementwise operation; β_2 controls the exponential decay rate for the moment estimates; and ϵ is a small constant that is inserted in the denominator in order to provide numerical stability. Typical values for β_2 and ϵ are, respectively, 0.9 and 1e−8.

$$S_{dw} = \beta_2 \cdot S_{dw} + \left(1 - \beta_2\right) dw^2 \tag{8}$$

$$S_{db} = \beta_2 \cdot S_{db} + \left(1 - \beta_2\right) db^2 \tag{9}$$

$$w = w - dw \cdot \frac{\alpha}{\sqrt{S_{dw}} + \epsilon} \tag{10}$$

$$b = b - db \cdot \frac{\alpha}{\sqrt{S_{db}} + \epsilon} \tag{11}$$

The Adam algorithm incorporates the ideas of GDM and RMSprop, and it is a robust method which is simple to implement and has shown to be consistently superior when compared with several other gradient-based optimization methods (Kingma and Ba 2015; Dozat 2016). The Adam algorithm first initializes V_{dw}, V_{db}, S_{dw}, and S_{db} to zero and then computes (8)–(9) and (12)–(13). After determining the new V_{dw}, V_{db}, S_{dw}, and S_{db} values, the model computes

(14)–(15), where t is the iteration number of the algorithm ($t \geq 1$). Finally, w and b are updated as in (16)–(17).

Equations (12)–(13) are very similar to (4)–(5), both are exponentially weighted averages, but (12)–(13) are a decaying mean, and (4)–(5) are a decaying sum. Actually, Eqs. (12)–(13) could be used substituting (4)–(5) in the GDM. Equations (14)–(15) are responsible to perform a bias correction. In the early stages of the algorithm, since fewer values have been accumulated in V_{dw}, V_{db}, S_{dw}, and S_{db}, the algorithm makes an underestimation of the expected values for these parameters. It is important to notice that as the algorithm evolves, the denominator of the algebraic expressions in (14)–(15) becomes one, and the bias correction step is no more needed. A bias correction can also be performed in the GDM and RMSprop, but it is not a common practice reported in the literature.

Typical values for β_1, β_2, and ϵ are 0.9, 0.999, and 1e−8, respectively.

$$V_{dw} = \beta_1 \cdot V_{dw} + \left(1 - \beta_1\right) dw \tag{12}$$

$$V_{db} = \beta_1 \cdot V_{db} + \left(1 - \beta_1\right) db \tag{13}$$

$$\widehat{V_{dw}} = \frac{V_{dw}}{\left(1 - \beta_1^t\right)}, \quad \text{and} \quad \widehat{V_{db}} = \frac{V_{db}}{\left(1 - \beta_1^t\right)} \tag{14}$$

$$\widehat{S_{dw}} = \frac{S_{dw}}{\left(1 - \beta_2^t\right)}, \quad \textit{and} \quad \widehat{V_{db}} = \frac{S_{db}}{\left(1 - \beta_2^t\right)} \tag{15}$$

$$w = w - \alpha \cdot \frac{\widehat{V_{dw}}}{\sqrt{\widehat{S_{dw}}} + \epsilon} \tag{16}$$

$$b = b - \alpha \cdot \frac{\widehat{V_{db}}}{\sqrt{\widehat{S_{db}}} + \epsilon} \tag{17}$$

Activation functions

In each hidden layer, after making a linear combination of the previous inputs, it is important to apply a nonlinearity to the result; otherwise, the overall capacity of the ANN to identify nonlinear behaviors would be nullified. There are several options of nonlinear functions that can be used, but the logistic sigmoid activation function (σ), hyperbolic tangent (tanh), and rectified linear unit (ReLU) are among the most well-studied (Goodfellow et al. 2016).

The logistic sigmoid activation function can be mathematically described as (18). Although extensively used in the past, in the last few years, this function is becoming less commonly used in the hidden layers of new ANN models as several papers have reported better results while using the tanh or ReLU (Glorot and Bengio 2010; Glorot et al. 2011, 2014; Maas et al. 2013). Equations (19) and (20) describe the tanh and ReLU activation functions, respectively.

According to (Wan et al. 2013) one of the main drawbacks of the sigmoid and *tanh* activation functions is that they saturate when x becomes very positive or negative, and when this happens the gradients become too small slowing down the learning process. Compared to the logistic sigmoid function the tanh has mean zero and is symmetric about the origin, and these properties make this activation function more likely to produce outputs that are on average closer to zero what helps the gradient descent algorithm (LeCun et al. 1998). If the output of an activation function is always positive (logistic sigmoid or ReLU) or always negative, the weight updates of each node can only all increase or all decrease together at each epoch, potentially slowing down the optimization (LeCun et al. 1998).

Finally, the ReLU activation function is not zero-centered and suffers from a problem called dying ReLU (Géron 2017), and this is observed during training when the sum of a specific neuron's input is negative. When this happens, this neuron will only output zero for the remaining of the training since the gradient of the ReLU function in $x < 0$ is zero. Despite these negative characteristics, the ReLU function has shown an excellent performance in several applications. Its simple formulation allows faster gradient calculations and the dying ReLU problem has shown not to be much prevailing when the weight initialization and the learning rate are properly chosen (Géron 2017). Additionally, some alternatives such as LeakyRelu (Xu et al. 2015) and ELU (Clevert et al. 2015) have been proposed in order to solve the dying ReLU problem and enhance the learning performance, but these functions also have pros and cons and are not used as often as the ReLU currently is.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{18}$$

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{19}$$

$$\mathrm{ReLU}(x) = \max(0, x) \tag{20}$$

Weight initialization

In order to train ANNs using algorithms based on the gradient descent method, it is crucial to properly initialize the model weights. These weights need to be initialized randomly and have an optimum scale, not too close neither too far from zero. A random initialization assures symmetry breaking in the

training phase (Goodfellow et al. 2016). Weights of the same layer would be updated identically during the backpropagation phase if they were initialized with the same value, making the ANN maybe even less effective than a linear regression model in that case.

Additionally, it is important to initialize the ANN weights with proper magnitudes in order to avoid the vanishing/exploding gradient problem (Schmidhuber 2015). Weights initialized with values very close to zero can potentially vanish the information that arrives from the input layer to the output layer ending up with extremely small output values in the first iterations and consequently small gradients. On the other hand, the initialization of the ANN weights with values far from zero may increase the output values severely, ending up with excessive large gradients. Overall, the vanishing/exploding gradient problem decreases the training speed and is exacerbated with the ANN depth. Actually, this is the main reason why it took longer for deep learning to start flourishing when compared to other shallow neural network models (Géron 2017).

Before the development of more sophisticated techniques (Glorot and Bengio 2010), weights were usually initialized to follow a uniform random distribution with zero mean and standard deviation one (Géron 2017). Although this initialization method can provide symmetry breaking in the ANN training, it does not properly scale the ANN weights what often leads to the vanishing/exploding gradient problem.

The work of (Glorot and Bengio 2010) proposed a new method to perform weight initialization in ANNs. The main idea was to control the variance of the input/output signals (forward propagation) and the variance of the gradients during the backpropagation phase so that one could avoid that input/output signals and gradients get exponentially large or small. When using the tanh activation function the authors in (Glorot and Bengio 2010) manage to prove that (21) needs to be satisfied in order to assure equal variance in the inputs and outputs of each ANN layer, and that (22) needs to be satisfied in order to assure equal variance in what concern the gradients of the backpropagation step. In (21) and (22), $\mathrm{Var}\left[w^{\ell}\right]$ and n_{ℓ} are, respectively, the variance of the weights and the number of neurons in the ℓ^{th} layer. Finally, it is only when n_{ℓ} is equal to $n_{\ell+1}$ that (21) and (22) can be simultaneous satisfied, so as a compromise, the authors decided to take the average of (21) and (22), arriving at the famous expression (23), which can be written in terms of uniform distribution (24) still satisfying the premises of the original proof presented in (Glorot and Bengio 2010).

$$\mathrm{Var}\left[w^{\ell}\right] = 1/n_{\ell} \tag{21}$$

$$\mathrm{Var}\left[w^{\ell}\right] = 1/n_{\ell+1} \tag{22}$$

$$\mathrm{Var}\left[w^{\ell}\right] = 2/\left(n_{\ell} + n_{\ell+1}\right) \tag{23}$$

$$U\left(-\sqrt{\frac{6}{n_{\ell}+n_{\ell+1}}}, \sqrt{\frac{6}{n_{\ell}+n_{\ell+1}}}\right) \tag{24}$$

Still following the derivation proposed in (Glorot and Bengio 2010), it is possible to arrive at proper weight initializations for the sigmoid (25) (Bengio 2012) and ReLU (26) (He et al. 2015) activation functions.

$$U\left(-4\sqrt{\frac{6}{n_{\ell}+n_{\ell+1}}}, 4\sqrt{\frac{6}{n_{\ell}+n_{\ell+1}}}\right) \tag{25}$$

$$U\left(-\sqrt{2}\sqrt{\frac{6}{n_{\ell}+n_{\ell+1}}}, \sqrt{2}\sqrt{\frac{6}{n_{\ell}+n_{\ell+1}}}\right) \tag{26}$$

With respect to the biases, they are usually initialized to zero since they do not contribute to symmetry breaking or training speed as the weight's initialization does (Witten et al. 2017).

Cost function and regularization

The training of complex ANN models with a small amount of data often leads to overfitting. However, this problem can be minimized in several cases by using some regularization techniques. Overfitting is a problem that happens when the ANN is too closely fit to a limited dataset. When this happens, the ANN has problems in generalizing its predictive ability to inputs/outputs that do not belong to the set which it has been trained. One way of solving this issue is by increasing the amount of data used during training. However, in several cases, it is difficult if not practically impossible, to obtain more data (Géron 2017; Glorot et al. 2014).

Overfitting is a combination of two factors, training data size, and ANN complexity. If an ANN is very large with many neurons and hidden layers, but the training dataset is small compared to the ANN complexity, the model tends to have an excessive number of degrees of freedom (Hinton et al. 2012c). In this situation, it is likely that the optimization model explores this over-parameterization to overfit the training dataset when none regularization technique is incorporated during learning (Glorot et al. 2014; Hinton et al. 2012c).

Three commonly used regularization techniques are ℓ_2 regularization, dropout (Glorot et al. 2014), and early stopping (Goodfellow et al. 2016; Géron 2017). In the ℓ_2 regularization, the square of the ℓ_2 norm of the weight matrixes (w_2^2) is used in the objective function as in (27), where the first term in the sum penalizes the difference between the expected and

estimated value, and λ is a regularization hyperparameter that has to be tuned (Goodfellow et al. 2016; Géron 2017).

In the dropout regularization, neurons are randomly turned off during training; this means that the weights and biases of these turned-off neurons are not considered during the epoch optimization. Because this method is constantly changing the ANN topology, it indirectly reduces the ANN capacity to overfit the training data (Glorot et al. 2014). In practice, this algorithm only needs to be zero the output of a specific neuron in order to assure that it will not interfere in the forward step neither have its parameters updated in the backward step. In a specific layer, while assuming a probability k_{prob} of keeping an individual neuron active, it is important to rescale the output of the reaming active neurons to ensure that the expected value of the turned-off neurons remains the same at each iteration, this is done by multiplying the output of each neuron by $1/k_{\text{prob}}$. Finally, the early stopping technique basically stops the ANN learning process right after the error in the development set starts increasing, giving signs of overfitting (Géron 2017).

In this work it was used the ℓ_2 or dropout regularization with early stopping. In the ℓ_2 (dropout) regularization the process of tuning the hyperparameter λ (k_{prob}) can be difficult, traditionally leading to several λ (k_{prob}) choices, where overfitting is still present. From this perspective, if the early stopping method is applied together with the ℓ_2 (dropout) regularization, the early stopping ends up interrupting simulations as they start presenting increases in the development set error, what helps to save computational time. Also, the cost function C was investigated in two different formulations (28)–(29), where $y^{(i,t)}$ ($\hat{y}^{(i,t)}$) is the estimate (expected value) for the tth day of forecast in the ith historical example. It is important to notice that (27) is only used as objective function for the ℓ_2 regularization method, while in the dropout method the objective function is defined directly by C.

Cost function (29) is the mean square error (MSE), where the difference between the estimated and expected streamflow is divided by σ_0, variance of the output variables, in order to normalize the error and disaggregate the metric from the streamflow scale.

$$J = C + \frac{\lambda}{2} w_2^2 \tag{27}$$

$$C = \frac{1}{m \cdot n} \sum_{i=1}^{n} \sum_{t=1}^{m} \left(\frac{y^{(i,t)} - \hat{y}^{(i,t)}}{\hat{y}^{(i,t)}} \right)^2 \tag{28}$$

$$C = \text{MSE} = \frac{1}{m \cdot n} \sum_{i=1}^{n} \sum_{t=1}^{m} \left(\frac{y^{(i,t)} - \hat{y}^{(i,t)}}{\sigma_0} \right)^2 \tag{29}$$

Performance metrics

Different error metrics can be used to evaluate the performance of streamflow forecasting models. Since each metric has its own distinct characteristics, being more sensitive to certain error patterns, many authors have sown the importance of choosing multiple metrics while comparing different streamflow forecasting models (Moriasi et al. 2007; Legates and McCabe 1999).

This work considers error metrics (29)–(32). As previously explained, Eq. (29) is the mean square error (MSE). Equation (30) computes the mean absolute percentage error (MAPE) considering each day of forecast individually, whereas (31) computes the MAPE for the average streamflow ($\overline{y^{(i)}}$) in each example i. Equation (32) is the Nash–Sutcliffe coefficient for the average streamflow $\overline{y^{(i)}}$.

The NSE coefficient can vary between $-\infty$ and 1 where a NSE = 1 corresponds to a perfect fit, NSE = 0 means that the model prediction is as accurate as the mean of the observed data, and NSE ≤ 0 means that the model is a worse predictor than the mean of the observed data (Legates and McCabe 1999).

The largest disadvantage of error metrics such as MSE and NSE that use the square of the difference between the estimated and observed values, is that they tend to be oversensitive to peak flows and are not very sensitive to systematic errors during low flow periods (Legates and McCabe 1999). On the other hand, the MAPE tends to assume the opposite characteristics of MSE and NSE; the MAPE is more sensitive to systematic errors and less sensitive to peak flows.

$$\text{MAPE} = \frac{100}{m \cdot n} \sum_{i=1}^{n} \sum_{t=1}^{m} \left| \frac{y^{(i,t)} - \hat{y}^{(i,t)}}{\hat{y}^{(i,t)}} \right| \% \tag{30}$$

$$\text{MAPE}_a = \frac{100}{n} \sum_{i=1}^{n} \left| \frac{\overline{y^{(i)}} - \overline{\hat{y}^{(i)}}}{\overline{y^{(i)}}} \right| \% \tag{31}$$

$$\text{NSE} = 1 - \frac{\sum_{i=1}^{n} \left(\overline{y^{(i)}} - \overline{\hat{y}^{(i)}} \right)^2}{\sum_{i=1}^{n} \left(\overline{\hat{y}^{(i)}} - \mu_0 \right)^2} \tag{32}$$

ANN input variable selection

Even though the input variable selection (IVS) process plays an important role in the ANNs model performance (May et al. 2011; Taormina and Chau 2015), some works still fail to provide a more detailed IVS investigation. The incorporation of redundant information in the ANN inputs

increases the number of local optimal solutions making the optimization algorithms more likely to converge to an unsatisfactory result. Also, the presence of irrelevant inputs adds noise into the model, what may disguise relevant input–output characteristics (May et al. 2011; Taormina and Chau 2015). Overall, both redundant and irrelevant information increase the computation efforts required during training and decrease the generalization capacity of the model.

This work follows an IVS procedure where the analysis of the correlation coefficients between the input/output is used to reduce the number of irrelevant information, and the analysis of the correlation coefficients between the input variables themselves is used to reduce redundancy. Arguably, IVS methods that are based on the Pearson correlation are among the most used for ANNs (May et al. 2011) that is not different in streamflow forecasting studies (Yaseen et al. 2015, 2016; Tongal and Booij 2018).

However, as the Pearson correlation gives a measure of the linear association between two variables, it may not properly sign the importance of relevant nonlinear interactions (Galelli et al. 2014). This is most likely not an issue in what concern the interaction between input/output streamflows since there is a significant linear correlation between each other. However, this may be an issue while analyzing the interaction of daily historical rainfall and daily forecasted (output) streamflow. This would happen because the Pearson correlation between these variables tends to be traditionally small if no data pre-processing approach is performed such as accumulate the rainfall through time periods or compute a weighted average rainfall (Bravo et al. 2009; Egawa et al. 2011).

Finally, before proceeding to the next sections where different IVS strategies are going to be presented, it is helpful to make the following definition. For a specific hydro plant $\mathcal{X}$, the first set of plants immediately upstream of plant $\mathcal{X}$ is going to be called the 1st level of upstream plants. Similarly, the 2nd level of upstream plants will correspond to all hydro plants that are immediately upstream of each plant of the 1st upstream level. Expanding the definition, the nth level of upstream hydro plants corresponds to all plants immediately upstream of each plant of the $(n-1)$th level. Figure 2 illustrates an example in a cascade system with different levels of upstream plants; in this case, *Ilha Solteira* and *Três Irmãos* are in the second level of upstream plants for *Porto Primavera*.

Also, the streamflow variables mentioned in this section and used throughout this work refer to natural water flow; that is, the flow that could be measured if the water was moving freely without the interference of dams.

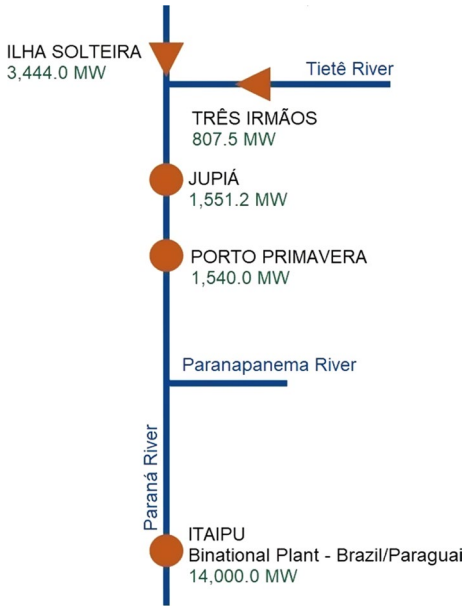

Fig. 2 Exemplifying levels of upstream hydro plants

Input variable selection—streamflow

Figure 3 presents a flow diagram of the streamflow IVS process. First, it is defined the forecasted output streamflow variables (s_O and D_{OS}), the set of input candidates, that is, which streamflow stations are going to be considered for a specific ANN model (In_s), and how many days of input streamflow are going to be investigated (D_{IS}). In this paper, the IVS considers as input candidates, all streamflow stations from the third downstream plant up to the 10th level of upstream hydro plants, and it is investigated 30 days of lag for input streamflow with 14 days of forecast (output layer). The previous constraints concerning the streamflow stations and days of lag were arbitrarily defined. However, after a proper analysis of the input candidates, if the results indicate that the ANN performance could benefit from a larger number of streamflow stations or days of lag, the IVS process can be repeated with more adequate intervals.

The number of nonrelevant input information is controlled by eliminating input candidates that do not assume correlation coefficients higher than $C_{IS/OS}$ with any of the output variables. After that, from the remaining input candidates, the streamflow stations that have correlation coefficients higher than C_{IS} are considered redundant. In this case, the station with the highest average correlation with the output variables is preserved and the other redundant stations are eliminated.

The optimal selection of $C_{IS/OS}$ and C_{IS} is made through a dynamic procedure, where random values of $C_{IS/OS}$ and C_{IS} are proposed between a pre-defined range and the correspondent input/output set is used in the training of an ANN

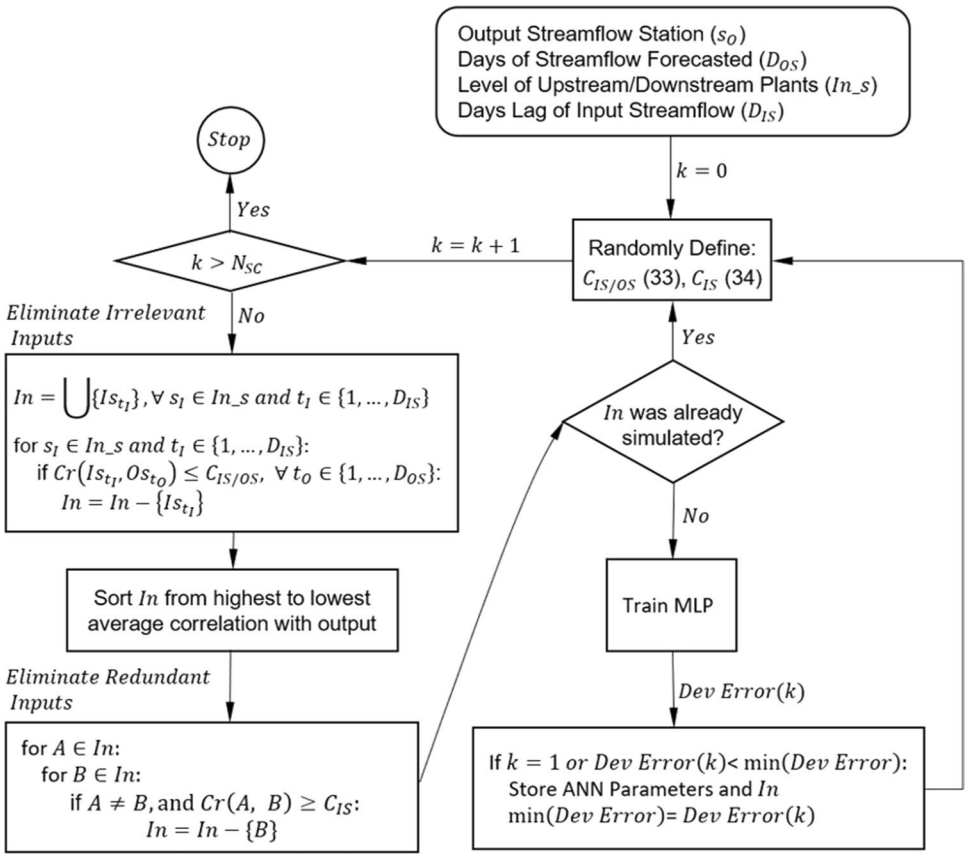

Fig. 3 Flow diagram IVS—streamflow

model. In this algorithm, after a fixed number of iterations (N_{SC}) the best input set is selected as the one with the lowest development error. In order to save computational time, if the $C_{IS/OS}$ and C_{IS} selection generates a set of input variables previously simulated, the algorithm then skips this ANN training and goes to the next random values for $C_{IS/OS}$ and C_{IS}.

In this work, $C_{IS/OS}$ and C_{IS} are proposed using (33) and (34), respectively, where $U_D(a, b)$ is a discrete uniform distribution in the interval [a, b]. Initially, the limits for these variables are arbitrarily defined. However, after some simulations, new insights about better limits can be draw, and in this case, one can re-write (33) and (34) and proceed with a new IVS analysis. This coarse-to-fine search was not performed in this work due to simulation time constraints.

$$C_{IS/OS} = 0.4 + U_D(0, 5) \cdot 0.1 \tag{33}$$

$$C_{IS} = 0.8 + U_D(0, 4) \cdot 0.05 \tag{34}$$

Input variable selection—rainfall

Figure 4 is a flow diagram that explains the IVS process that includes rainfall and streamflow information. The IVS process that includes only rainfall information can be directly inferred by neglecting the elements depicted in red in this figure. First, it is defined a set of input candidates (In_s, D_{Is}, In_p, D_{IR}) and output variables (s_O, D_{OS}). In this work it is investigated 30 days of lag of input daily/accumulated rainfall and streamflow for 14 days of forecasted streamflow. After that, the model of Fig. 3 is used to determine the optimal set of streamflow inputs (In_S^*). In this work, the term daily rainfall is used to refer to the total rainfall observed in a single day and the term accumulated rainfall to refer to the sum of multiple daily rainfalls.

If one decides to accumulate rainfall information, the number of accumulated rainfall days for a specific rainfall station is the one that maximizes the average correlation between the first day of accumulated rainfall and forecasted streamflow. It is important to mention that the same rainfall station may use a different number of days to accumulate rainfall depending on the hydro plant (s_O) analyzed.

The next steps of this IVS process follow a similar procedure as explained for Fig. 3. The number of nonrelevant input information is controlled by eliminating input candidates that do not assume correlation coefficients higher than $C_{IR/OS}$ with any of the output variables. After that, from the remaining input candidates, the rainfall stations that have correlation coefficients higher than C_{IR} are considered

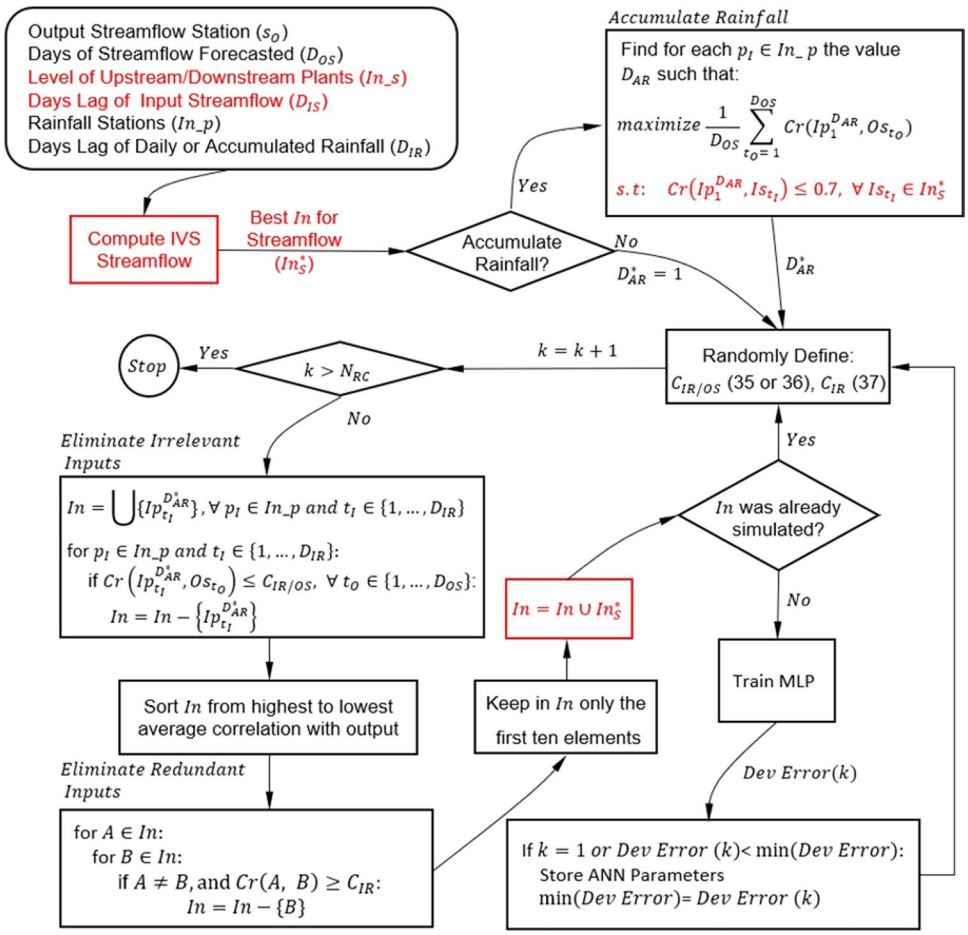

Fig. 4 Flow diagram IVS—streamflow and/or rainfall

redundant. In this case, the station with the highest average correlation with the output variables is preserved and the other redundant stations are eliminated. After reducing redundancy and nonrelevant information it is still possible that a large number of input candidates are considered, what could increase expressively the simulation time. To minimize this problem, only a maximum of ten rainfall stations are kept and the other are eliminated. These maximum ten remaining stations are the ones with the highest average correlation with the output variables.

The redundancy between daily rainfall and streamflow inputs was not controlled since these variables have low correlation coefficients (on average 0.1 with maximum values around 0.4). Accumulated rainfall and input streamflow have higher correlation coefficients between each other; thus, in order to control redundancy, the process of defining the number of accumulated rainfall days was constrained to outcome rainfall input candidates with correlation coefficient lower than 0.7 with the previously determined input streamflow variables.

In what concern the $C_{IR/OS}$ limits, they were proposed as (35) in the case of daily rainfall and as (36) in the case of accumulated rainfall; the C_{IR} limits are proposed as (37) in both cases. Both (35)–(36) and (37) were arbitrarily defined based on a previous analysis of the correlation coefficients involved.

$$C_{\mathrm{IR/OS}} = 0.2 + U_{\mathrm{D}}(0,2) \cdot 0.05 \tag{35}$$

$$C_{\mathrm{IR/OS}} = 0.4 + U_{\mathrm{D}}(0,4) \cdot 0.05 \tag{36}$$

$$C_{\mathrm{IR}} = 0.8 + U_{\mathrm{D}}(0,4) \cdot 0.05 \tag{37}$$

Results and discussion

Case study

The case study is focused on streamflow forecasting for hydropower plants in Brazil. Short-term forecasting studies are extremely important in Brazil since they are not only used in the operation of the electrical system but also in the definition of the electricity spot prices (Souza and Legey 2008; de Queiroz et al. 2007). These prices value the energy traded in the short-term energy market that nowadays

corresponds to about 30% of the electricity consumption in Brazil (ABRACEEL 2018) and moves billions of dollars each year.

The models trained in this work used historical natural streamflow and/or rainfall information from the Paraná Basin in Brazil. The historical streamflow data were obtained from the Brazilian National Water Agency (ANA 2019) and are available in daily discretization measured for each of the 55 hydro plants shown in Fig. 5. The historical rainfall data from CPTEC-INPE can be found in (CPTEC 2019) and combine satellite precipitation estimates with observed rainfall, this model is called MERGE (Rozante

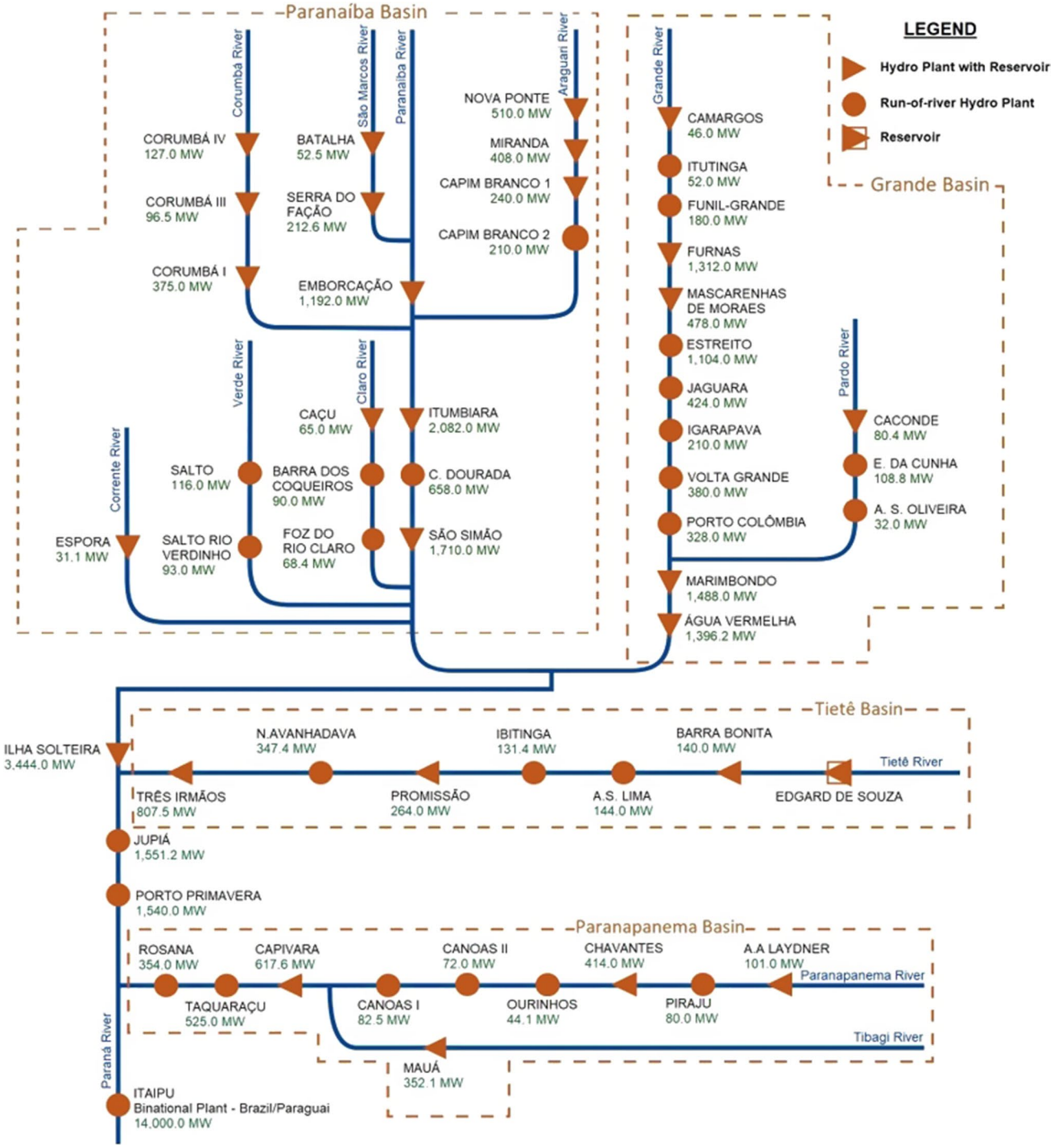

Fig. 5 Paraná basin—schematic of the hydro plants considered

et al. 2010), and its data are available in daily discretization and 0.1° grid resolution. The data used in this work span from January 1, 2000, up to December 31, 2016. The first 60% of the record is used as the training set (years 2000 to 2010), the next 20% for the development set (years 2010 to 2013), and the remaining 20% for the test set (years 2013 to 2016) of the ANNs.

All ANNs trained are fully connected, the number of neurons in each hidden layer is equal to the number of neurons in the input layer, and the number of hidden layers is fixed in three as in Tongal and Booij (2018), Bravo et al. (2009). Fine-tune the number of hidden layers and neurons per hidden layer was not pursued because this would imply a substantial increase in model training time. Instead, an over-parameterized model was adopted, with number of hidden layers high enough to overfit the training set, and providing freedom for the regularization models to control the generalization capacity of the neural networks (Goodfellow et al. 2016; Géron 2017).

Additionally, when nothing is specified, the ANN training stops if a maximum number of 30,000 epochs are reached ($\overline{\text{Epoch}} = 30,000$) or condition (38) is satisfied, where $E_{\text{dev}}(k)$ is the development set error for the kth epoch. It is important to mention that, independently of the number of epochs, the models will always store the weights and biases of the epoch with the lowest development set error. It is also important to mention that $E_{\text{dev}}(k)$ is computed using the element C of the cost function.

$$E_{\text{dev}}(k) - E_{\text{dev}}(k - 5000) > 0 \quad \forall k > 5000 \tag{38}$$

The optimal values for α, λ, and k_{prob} are investigated iteratively through a cross-validation process where different values for these hyperparameters are selected randomly and used in the ANN training. The hyperparameters α and λ are selected from a uniform distribution in a base 10 log-scale in order to assure equal probability while sampling values in very different scales say 10^{-2} and 10^{-6} (Goodfellow et al. 2016). On the other hand, k_{prob} is sampled directly from a uniform distribution because this hyperparameter is not as sensitive as α and λ are to small variations. In a determined ANN input configuration, for all α, λ, and k_{prob} values tested only the training with the lowest development set error is stored.

For all simulations presented in this section, the upper bounds for α and λ are 1e−2, and the lower bounds are 1e−5 and 1e−6, respectively, (39)–(40). For the hyperparameter k_{prob} it is investigated the interval (0.2, 1] (41). Additionally, for the sake of simplicity, the number of random α, λ, and k_{prob} values tested is going to be denoted as N_{Hyp}, and the number of random $C_{\text{IS/OS}}$ ($C_{\text{IR/OS}}$) and C_{IS} (C_{IR}) correlation coefficients tested is going to be denoted as N_{SC} (N_{RC}). The intervals of search for α, λ, and k_{prob} were kept wide in order to be applied in the different algorithms and methods investigated in this work. Also, the consistency of the results provided by these search intervals was checked at each model simulation to guarantee that no model would lose performance due to inadequate hyperparameter sampling.

$$\alpha = 10^{-U(2,5)} \tag{39}$$

$$\lambda = 10^{-U(2,6)} \tag{40}$$

$$k_{\text{prob}} = U(0.2, 1) \tag{41}$$

For the simulations further presented in this section, the computational experiments were performed using a PC with an i7-7700 k CPU (4 cores, 4.2 GHz), 16 GB RAM, and a GPU NVIDIA GTX 1070 (8 GB). The TensorFlow (TensorFlow 2019) framework was used with GPU parallelization.

The performance of different ANN techniques

It is investigated ANNs for forecasting streamflow 14 days ahead in daily discretization. Five hydro plants are explored, namely, *Furnas*, *I. Solteira*, *Itaipu*, *Itumbiara*, and *Promissão* (Fig. 5). The models for *I. Solteira* and *Itaipu* represent critic simulations. For *I. Solteira*, the water of two large basins discharges in this plant, with a significant number of upstream hydro plants and rivers. For *Itaipu* the scenario is even more complex since the water of all other 54 hydro plants simulated discharge in this plant. The *Furnas* and *Itumbiara* plants are extremely important for their basins but have very different upstream configurations. Finally, *Promissão* represents a small hydro plant with smaller inflows when compared to the other plants mentioned above.

For these models, N_{Hyp} is fixed at the value 30, and it was used three hidden layers as in (Tongal and Booij 2018; Bravo et al. 2009). Each ANN uses as input 30 days of lag with respect to the streamflow information from the first downstream plant up to the 5th level of upstream hydro plants.

In the applications presented in this section it was not considered an IVS process, since the large number of simulations would increase significantly the computational time, when compared to an approach where the best input configuration is not investigated in a cross-validation process. As the main goal of this application is to compare the performance of different ANN techniques for streamflow forecasting, the use of a fixed input configuration for each hydro plant satisfies the overall goal.

Comparison of activation functions, optimization, and weight initialization

The objective of this section is to evaluate the contributions of different activation functions, weight initialization, and

optimization methods while training ANN models to perform streamflow forecasting. It is investigated the ReLU, the *tanh*, and the logistic sigmoid activation functions using three different optimization algorithms, namely, Adam, RMSprop, and GDM. The weight initialization is investigated in two different variants, one that uses the standard normal distribution, and another that uses the methods described in (24)–(26) that attempts to preserve the variance between input/output during training, this last one called optimal weight initialization (OWI). It is important to notice that for the OWI each activation function has its own weight initialization distribution.

Table 1 shows the results for the simulations mentioned above. Columns "MSE" and "MAPE" represent, respectively, the average of metrics (29) and (30) for all hydro plants considered. In this case, it is used only the performance of the best ANN model for each hydro plant. The column "Avg Epoch" represents the average number of epochs during training, and the column "Total Wall Time" represents the sum of the wall time during training. In these two last columns it is considered all simulations performed for each hydro plant while using a specific set of techniques (weight initialization, optimization algorithm, and activation function), not only the ANNs with the best performance. For more information regarding other error metrics such as (31)–(32) and individual hydro plant performance, please refer to Supplementary Note 1.

The results from Table 1 bold highlight the importance of a proper weight initialization. From the Adam models that do not use an OWI to the ones that use, there is a reduction in the simulation time of at least 52%. The stability of the models also benefited expressively from the OWI methods; for the Adam/ReLU model without OWI, all hydro plants faced the exploding gradient problem (Géron 2017), leading to model divergence in the first iterations.

According to Table 1 the OWI/Adam models have a very consistent performance being the best or close to the best model for all metrics. Also, for the models that use OWI, the tanh activation function shows a better performance in terms of accuracy when compared to the ReLU and Sigmoid. Although the GDM/tanh model had the worst performance regarding computational time and average number of epochs, it still performed well in terms of the error metrics analyzed. In terms of individual hydro plant performance and other error metrics such as "$MAPE_a$" and "NSE" (Supplementary Note 1) the model OWI/Adam/tanh still shows the most promising results.

Comparison of cost functions and regularization techniques

This section evaluates the performance of the ANN models while using cost functions (28)–(29) together with the ℓ_2 or dropout regularization techniques. It used the Adam optimization algorithm with the tanh activation function and variable initialization described by Eq. (30).

The results are presented in Table 2. In the column "Configuration," *M2* represents cost function (28) and *R2* cost function (29), the second element after the dash represents the regularization technique; the columns "MSE," "MAPE," "$MAPE_a$," and "NSE" represent the average of each individual error metric (29)–(32) for the hydro plants investigated; in these columns it is considered only the best ANN model for each hydro plant. Still regarding Table 2, the column "Wall Time" is the sum of the total wall time for all simulations performed in the ANNs investigated. For more information and details about individual hydro plants, please refer to Supplementary Note 2.

Based on the results of Table 2, it is possible to notice that the dropout regularization technique presented a better performance when compared with the ℓ_2 method. It is

Table 1 Performance for different weight initializations, optimization algorithms, and activation functions

Weight initialization (WI)	Optimization algorithm	Activation function	Error test set		Error dev set		Avg epoch	Total wall time (h)
			MSE	MAPE	MSE	MAPE		
Standard normal distribution	Adam	ReLU	Simulations Diverged in the First Iterations					
		Tanh	0.2710	17.99	0.4012	15.14	13,464	11.20
		Sigmoid	0.2605	18.11	0.3859	**14.63**	10,956	10.61
Optimal weight initialization (OWI)	Adam	ReLU	0.2729	18.71	0.3821	14.92	2598	5.09
		Tanh	**0.2574**	17.76	0.3762	14.78	**1668**	4.26
		Sigmoid	0.2577	18.43	0.3883	14.72	1680	**4.24**
	RMSprop	ReLU	0.2668	19.52	**0.3700**	15.15	2700	5.10
		Tanh	0.2647	18.21	0.3941	15.22	2516	4.67
		Sigmoid	0.2647	20.33	0.3945	15.88	4024	5.35
	GDM	ReLU	0.2809	19.37	0.3959	15.50	15,862	14.37
		Tanh	0.2605	**17.64**	0.3785	15.05	19,488	16.48
		Sigmoid	0.2749	18.92	0.4011	15.11	15,918	13.76

Table 2 Performance for different cost functions and regularization

Configuration	Test set error				Dev set error				Wall time (h)
	MSE	MAPE	$MAPE_a$	NSE	MSE	MAPE	$MAPE_a$	NSE	
*R*2-dropout	0.2517	24.51	20.58	0.677	**0.3269**	21.45	18.83	**0.772**	4.90
M2-dropout	**0.2332**	**17.30**	**15.56**	**0.736**	0.3618	**14.53**	**13.23**	0.737	5.38
$R2-\updownarrow_2$	0.2789	31.06	27.54	0.626	0.3403	21.43	18.81	0.760	**3.88**
$M2-\updownarrow_2$	0.2574	17.76	16.09	0.699	0.3762	14.78	13.56	0.724	4.26

interesting to notice that this difference in performance is less significant for models that use the *M*2 cost function than for the models that use the *R*2 cost function.

In what concern the test set, the *M2*-dropout configuration had the best average performance for all error metrics. In terms of the dev set error, the metrics MSE and NSE performed better in the *R*2 cost function; on the other hand, the metrics MAPE and $MAPE_a$ performed better in the *M*2 cost function. In terms of individual hydro plants (Supplementary Note 2) a few points are interesting to mention: the *NSE* tends to follow the *MSE* behavior; with a few exceptions, the *M2*-dropout configuration presents the most stable performance, being the best or near the best model for all metrics in the test set. In general terms, the individual hydro plant-level performance is similar to the one presented in Table 2.

The behavior of the dev set, where the metrics MSE and NSE performed better in the *R*2 cost function and the metrics MAPE and $MAPE_a$ perform better in the *M*2 cost function, was already expected. The cost function *R*2 carries more similarities with the metrics MSE and NSE; on the other hand, the cost function *M*2 is more similar to the metrics MAPE and $MAPE_a$. As the dev set participated in a cross-validation process, it is common that the error metrics that carry more similarities with the cost function are beneficiated. In this context, it is important to understand that the dev set error may not provide a "true" estimate of the model performance as the test set does.

In Table 2, between models $M2-\ell_2$ and *M2*-dropout is possible to notice an increase of about 26% in the total simulation time. Given that these two models have similar performance, this time difference must be taken into consideration while deciding for the use of the ℓ_2 or dropout regularization technique. Finally, it is also possible to notice, in the individual hydro plant level, that ANNs trained using the *R*2 cost function tend to be much more sensitive to sudden increases in flow, whereas ANNs trained using the *M*2 cost function are more stable, with small errors during low flow conditions but higher errors during sudden increases in streamflow. In order to illustrate this case Fig. 6 shows the estimated and measured streamflow for the *Furnas* hydro plant during the test set while using the *R*2 (Fig. 6a) and *M2* (Fig. 6b) cost functions and dropout regularization. For the hydrographs of other hydro plants please refer to Supplementary Note 2.

Comparison of ANNs and Brazilian ISO streamflow model

In this section, ANNs were trained to forecast streamflow 14 days ahead for all 55 hydro plants of Fig. 5 and the results were compared with the performance of the models currently used by the Brazilian ISO from 2014 up to 2016 (ONS 2014, 2015, 2016). In terms of data and models used during training of the ANNs, only streamflow information is used as input, an IVS model is applied as described in Fig. 3, the

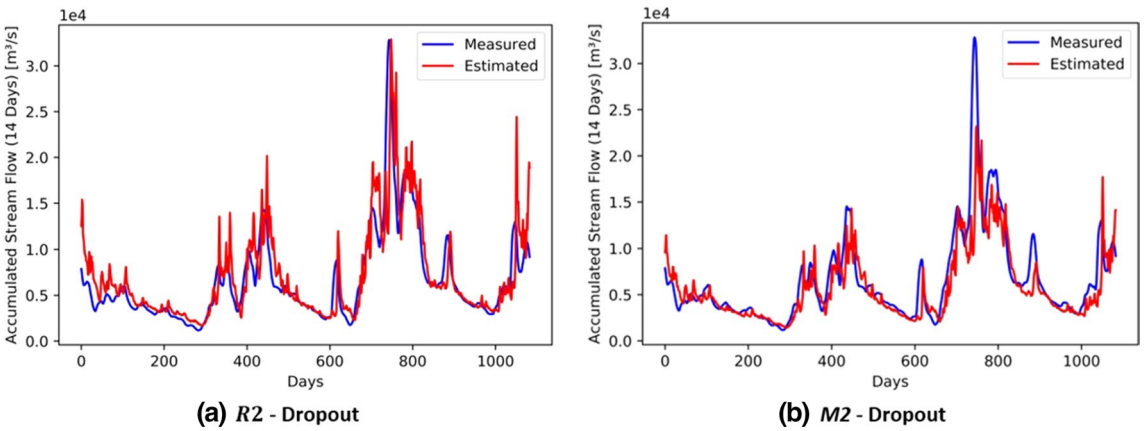

Fig. 6 Hydrograph—ANN estimate versus measured stream flow (*Furnas Hydro Plant)*

OWI/Adam/tanh model is used with N_{Hyp} fixed in 20, N_{SC} fixed in 20, and the number of hidden layers equals to three as in (Tongal and Booij 2018; Bravo et al. 2009). Also, it used $M2$ (28) cost function with the dropout regularization technique.

The Brazilian ISO is continually evaluating the performance of its hydrological models in order to implement and propose changes to increase accuracy. Nowadays, for the *Paraná Basin*, mainly two models are used, namely soil moisture accounting procedure (SMAP) and Previvaz (ONS 2017). The SMAP (Lopes et al. 1982; Lopes and Montenegro 2017) is an empirical deterministic model that considers three mathematical reservoirs to represent surface, soil and groundwater store and uses as input data rainfall and evaporation dividing the overland flow based on the concepts of the soil conservation service model (Mishra and Singh 2003). The Previvaz is a univariate stochastic model that uses the historical streamflow of each hydro plant to evaluate different linear time series models such as AR, ARMA, PAR, and PARMA (Souza et al. 2010; Box et al. 2015; Rasmussen et al. 1996); from these different models it is selected for each hydro plant the alternative with the best accuracy.

Figure 7a shows a boxplot of the difference between the average forecast errors ($MAPE_a$) of the Brazilian ISO and the ANN models from 2014 up to 2016. Figure 7b shows a similar information of Fig. 7a but comparing the NSE coefficient. In both figures a positive value means a better performance of the ANNs over the ISO models. Additionally, the errors $MAPE_a$ and NSE were chosen because they are the error metrics officially used and reported by the Brazilian ISO in the documents (ONS 2014, 2015, 2016).

Overall, it is possible to notice that the ANNs performed significantly better when compared to the models used by the Brazilian ISO. In terms of the $MAPE_a$ metric, the ANNs had lower forecasting errors for all hydro plants and years investigated (Fig. 7a). In terms of the Nash–Sutcliffe metric (Fig. 7b), the majority of the ANNs performed better than the Brazilian ISO during the years of 2015 (93%) and 2016 (82%), but in 2014 the ISO had a better performance in this metric for 72% of the hydro plants. However, it is interesting to notice that in 2014 the median and the third quartile presented in Fig. 7b are very close to zero, meaning that at least 25% of the ANN and ISO models had a similar performance in this period, with the ISO slightly outperforming the ANNs. For more information regarding individual hydro plant-level performance please refer to Supplementary Note 3.

Evaluating the historical rainfall as an input variable

This section evaluates the contributions of historical rainfall information as input for ANN models trained to perform streamflow forecasting 14 days ahead. It is investigated five hydro plants, namely, *Furnas*, *I. Solteira*, *Itaipu*, *Itumbiara*, and *Promissão*.

The simulations made in this section use the OWI/Adam/*tanh* model with N_{RC} fixed in 10, N_{Hyp} fixed in 20, and the number of hidden layers fixed in three as in (Tongal and Booij 2018; Bravo et al. 2009). Both daily and accumulated rainfall information are investigated using the IVS process described in Fig. 4. Also, $M2$ (28) and $R2$ (29) cost functions are investigated using the dropout regularization technique.

Table 3 presents the results for these simulations in terms of the MSE and MAPE. Other error metrics such as $MAPE_a$ and NSE are available in Supplementary Note 4.

Although the process of accumulate rainfall increased the average correlation between the input rainfall and output streamflow from 0.21 (daily rainfall) to 0.65 the results sign no clear advantage in applying this procedure. Additionally, the models that used both rainfall and streamflow information performed generally worse than the ones that used only

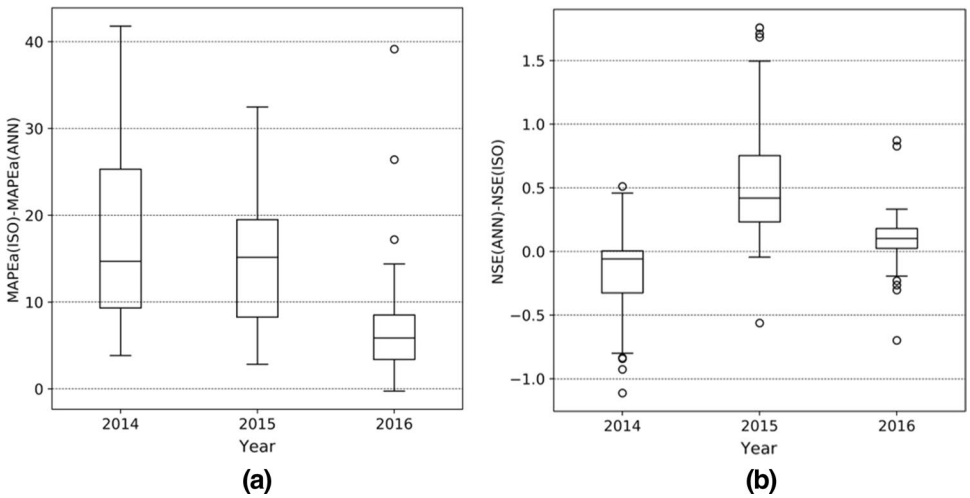

Fig. 7 Comparison of the forecasting error between the Brazilian ISO models and ANNs from 2014 to 2016—boxplot MAPE (**a**) and NSE (**b**)

Table 3 Contribution of the historical rainfall information

Cost function and regularization	Hydro plant	Test set error							
		Streamflow		Streamflow and accumulated rainfall		Streamflow and daily rainfall		Daily rainfall	
		MSE	MAPE	MSE	MAPE	MSE	MAPE	MSE	MAPE
*M*2-dropout	Furnas	**0.1511**	**21.20**	0.1539	23.57	0.1596	23.02	0.2370	57.97
	I. Solteira	**0.0694**	**11.39**	0.0744	14.06	0.0903	15.61	0.3431	43.27
	Itaipu	**0.2635**	**11.14**	0.3071	12.80	0.2897	12.35	0.5991	23.53
	Itumbiara	0.1039	**16.34**	**0.0989**	18.89	0.1083	18.94	0.1887	38.81
	Promissão	0.5603	**19.58**	**0.5377**	21.37	0.5509	20.07	0.7189	36.53
*R*2-dropout	Furnas	**0.1565**	**29.07**	0.1760	35.07	0.1842	36.17	0.4520	71.27
	I. Solteira	**0.0880**	**15.42**	0.1012	19.97	0.1181	20.66	0.4099	52.02
	Itaipu	**0.2205**	**10.89**	0.3042	13.94	0.2877	13.99	0.5422	23.03
	Itumbiara	**0.1207**	30.41	0.1568	**29.00**	0.1749	25.52	0.4020	70.13
	Promissão	0.4900	**27.05**	0.6423	38.91	**0.4827**	27.42	0.7133	39.77

streamflow. This intriguing result was already noticed in a few works (Akhtar et al. 2009; Aytek et al. 2008) and should be taken into consideration while developing new ANN models for short-term streamflow forecasting.

Still analyzing Table 3 (and Supplementary Note 4), it is possible to notice that some hydro plants slightly improved the MSE and NSE metric. On the other hand, these improvements are followed by the worsening of the MAPE metric, what shows that the ANN models are facing a certain difficulty in incorporating gains for these three metrics simultaneously. It is also interesting to notice that although the *R*2 cost function presents a more sensitive behavior to rapid increases in streamflow when compared to the *M*2 cost function, the historical rainfall information still does not provide a consistent contribution for the forecasting in the *R*2 models.

Conclusion

Due to the large number of decisions that have to be made while building an ANN model, it is a common practice to arbitrary select some of them, mainly the optimization algorithm, activation function, and weight initialization method. On the other hand, when multiple comparisons of different ANN algorithms/techniques are made, better ANN structures can be obtained, leading to improved forecasts and computational time savings.

In the case of streamflow forecasting, although the literature is diverse in terms of ANN models studied, it lacks comparisons among some aspects of each model. From this perspective, due to the large heterogeneity of the hydrological data used in each work, it is difficult to draw meaningful comparisons between papers that use different ANN algorithms/models, because most of them focus on specific configurations better suited for the application at hand. With the objective to contribute to a better understanding of the performance of different ANN models in streamflow forecasting studies, this paper evaluates various optimization models, activation functions, and weight initialization techniques in a large-scale system with 55 streamflow stations, each corresponding to a hydro plant from the *Paraná Basin* in Brazil. The results present relevant insights for those interested in applying the ANN theory in streamflow forecasting studies. Specially, we show that some ANN configurations perform more accurately to fast rises of streamflow. On the other hand, some models attenuate the forecasting of streamflow peaks giving preference to long-term low flow periods.

The paper also formalizes an input variable selection procedure using the correlation coefficients between input/output information in a cross-validation process where different limits of correlation coefficients are investigated together with other hyperparameters such as the learning rate and the regularization coefficient. The performance of the ANNs developed in this work is compared with the hydrological models used by the Brazilian ISO. The results show a strong superiority of the ANNs and provide theoretical support for the incorporation of this technique as a tool in the generation and trading processes performed by the Brazilian ISO and other electric power system agents. Finally, this paper evaluates the marginal contribution of the historical rainfall information while used together with historical streamflow in the ANN models. Although some hydro plants presented small benefits in accuracy while using this information, in general terms, the results show no clear advantage in incorporating this variable in the ANN training. This marginal influence of the historical rainfall information is likely due to the time horizon and hydrological basin explored. Nonetheless, it is interesting to notice that the ANN models overperformed the

empirical and conceptual models used by the Brazilian ISO with only streamflow information as its input.

Future works should evaluate the performance of other ANN techniques such as convolutional and recurrent when compared to the MLP models in the forecasting of streamflow for large interconnected hydro systems. Also, the benefits of using historical forecasted rainfall and temperature could be investigated using different weather forecast models such as the ones provided by (NOAA 2019) and (ECMWF 2019). Finally, it would be interesting that future works investigate new objective functions and optimization strategies that benefit the training of streamflow forecasting ANN models.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s13762-021-03565-y.

Acknowledgements The authors would like to express their gratitude to AES Tietê for the financial support of the ANEEL Strategic R&D Project 0064-1052/2017. The authors also would like to thank Eduardo H.S. Silva, Lívia M.P. Gazzi, Eliude P. Ferro, and Lanai Torres for technical discussions during the course of this project.

Authors' contribution All authors contributed to the study conception and design. Material preparation, data collection, and analysis were performed by VF and AdQ. The first draft of the manuscript was written by VF, and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding This research was funded by AES Tietê Brazil under the ANEEL Strategic R&D Project 0064-1052/2017.

Availability of data and material The data used in this research are publicly available on the internet.

Code availability Available upon request.

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

ABRACEEL—Brazilian Association of The Energy Traders (2018) Annual Report on the Abraceel Activities (**in portuguese**)

Aggarwal CC (2018) Neural networks and deep learning: a textbook. Springer, Berlin

Akhtar MK, Corzo GA, Andel SJ, Jonoski A (2009) River flow forecasting with artificial neural networks using satellite observed precipitation pre-processed with flow length and travel time information: case study of the Ganges river basin. Hydrol Earth Syst Sci 13:1607–1618

ANA—Brazilian National Water Agency (2019) Operational data from the reservoirs of the Brazilian electric system, viewed 10 January 2021. www.ana.gov.br (**in portuguese**)

Aytek A, Asce M, Alp M (2008) An application of artificial intelligence for rainfall-runoff modeling. J Earth Syst Sci 117:145–155

Bengio Y (2012) Practical recommendations for gradient-based training of deep architectures. Neural networks: tricks of the trade. Springer, Berlin, pp 437–478

Box GEP, Jenkins GM, Reinsel GC, Ljung GM (2015) Time series analysis forecasting and control, 5th edn. Wiley, London

Bravo JM, Paz AR, Collischonn W, Uvo CB, Pedrollo OC, Chou SC (2009) Incorporating forecasts of rainfall in two hydrologic models used for medium-range streamflow forecasting. J Hydrol Eng 14(5):435–445

Chen L, Singh VP, Guo S, Zhou J, Ye L (2014) Copula entropy coupled with artificial neural network for rainfall–runoff simulation. Stoch Environ Res Risk Assess 28:1755–1767

Clevert D, Unterthiner T, Hochreiter S (2015) Fast and accurate deep network learning by exponential linear units (ELUs). CoRR abs/1511.0

CPTEC—Center for weather forecasting and climate studies of the Brazilian National Institute for Space Research (2019) MERGE, viewed 10 January 2021. http://ftp.cptec.inpe.br/modelos/io/produtos/MERGE/

de Queiroz AR, Oliveira FA, Lima JWM, Balestrassi PP (2007) Simulating electricity spot prices in brazil using neural network and design of experiments. IEEE Lausanne Power Tech

de Queiroz AR, Lima LMM, Lima JWM, da Silva BC, Scianni LA (2016) Climate change impacts in the energy supply of the Brazilian hydro-dominant power system. Renew Energy 99:379–389

Dozat T (2016) Incorporating nesterov momentum into adam. In: International conference on learning representations (ICLR)

ECMWF—European Center for Medium-Range Weather Forecasts (2019) viewed 10 January 2021. www.ecmwf.int

Egawa T, Suzuki K, Ichikawa Y, Lizaka T, Matsui T, Shikagawa Y (2011) A water flow forecasting for dam using neural networks and regression models. IEEE Power and Energy Society General Meeting.

Eslamian S (2014) Handbook of Engineering hydrology: modeling, climate change, and variability. CRC Press, London

Faria VAD, de Queiroz AR, Lima LMM, Lima JWM (2018) Cooperative game theory and last addition method in the allocation of firm energy rights. Appl Energy 226:905–915

Galelli S, Humphrey GB, Maier HR, Castelletti A, Dandy GC, Gibbs MS (2014) An evaluation framework for input variable selection algorithms for environmental data-driven models. Environ Model Softw 62:33–51

Géron A (2017) Hands-on machine learning with Scikit-Learn and TensorFlow. O' Reily Media

Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. Proc AISTATS 9:249–256

Glorot X, Bordes A, Benigo Y (2011) Deep sparse rectifier neural networks. AISTATS 15:315–323

Glorot X, Bordes A, Benigo Y (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15:1929–1958

Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, New York

He K, Zhang X, Ren S, Sun J (2015) Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. ICCV

Hinton GE, Deng L, Yu D, Dahl G, Mohamed A, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath T, Kingsbury B (2012a) Deep

neural networks for acoustic modeling in speech recognition. IEEE Signal Process Mag 29:82–97
Hinton GE, Srivastava N, and Swersky K (2012b) Lecture 6a overview of mini–batch gradient descent. Coursera Lecture slides, viewed 10 January 2021. https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf
Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR (2012c) Improving neural networks by preventing co-adaptation of feature detectors. arXiv:1207.0580v1
Hsu K, Gupta HV, Sorooshian S (1995) Artificial neural network modeling of the rainfall-runoff process. Water Resour Res 31:2517–2530
Iha—International Hydropower Association (2018) Hydropower status report
Jo T, Hou J, Eickholt J, Cheng J (2015) Improving protein fold recognition by deep learning networks. Sci Rep 5:1–11
Kingma DP, Ba JL (2015) Adam: a method for stochastic optimization. In: International conference on learning representations (ICLR)
KİŞİ Ö (2007) Streamflow forecasting using different artificial neural network algorithms. J Hydrol Eng 12(5):532–539
Krizhevsky A, Sutskever I, Hinton G (2012) ImageNet classification with deep convolutional neural networks. In: NIPS
Kumar DN, Raju KS, Sathish T (2004) River flow forecasting using recurrent neural networks. Water Resour Manag 18:143–161
LeCun Y, Bottou L, Orr GB, Muller KR (1998) Efficient backprop. In: Neural networks, tricks of the trade, Lecture Notes in Computer Science LNCS 1524
Legates DR, McCabe JG (1999) Evaluating the use of goodness-of-fit measurements in hydrologic and hydroclimatic model validation. Water Resour Res 35:233–241
Li X, Shang W, Wang S (2018) Text-based crude oil price forecasting: a deep learning approach. Int J Forecast 35(5):1548–1560
Lima MMM, Popova E, Damien P (2014) Modeling and forecasting of Brazilian reservoir inflows via dynamic linear models. Int J Forecast 30:464–476
Lopes I, Montenegro AAA (2017) Hydrological process simulation at plot scale using the smap model in the semiarid 3:78–86
Lopes JEG, Braga BPF, Conejo JGL (1982) SMAP—a simplified hydrological model. In: Applied modeling in catchment hydrology Water Resources Publications, Littleton, pp 167–176
Maas AL, Hannun AY, Ng AY (2013) Rectifier nonlinearities improve neural network acoustic models. In: International conference on machine learning (ICML)
May R, Dandy G, Maier H (2011) Review of input variable selection methods for artificial neural networks artificial neural networks—methodological advances and biomedical applications. InTech
Mishra SK, Singh VP (2003) Soil conservation service curve number (SCS-CN) methodology. Springer, Netherlands
Moriasi DN, Arnold JG, Van MWL, Bingner RL, Harmel RD, Veith TL (2007) Model evaluation guidelines for systematic quantification of accuracy in watershed simulations. Trans Am Soc Agric Biol Eng 50:885–900
Morse G, Stanley KO (2016) Simple evolution optimization can rival stochastic gradient descent in neural networks. In: Proceedings of the genetic and evolutionary computation conference.
Najafabadi MM, Villanustre F, Khoshgoftaar TM, Seliya N, Wald R, Muharemagic E (2015) Deep learning applications and challenges in big data analytics. J Big Data 2:1–21
NOAA—National Centers for Environmental Information (2019), viewed 10 January 2021. <www.ncdc.noaa.gov>
ONS—Brazilian Independent System Operator (2014) Annual Report of Streamflow Forecasting 2014, viewed 10 January 2021. www.ons.org.br (**in portuguese**)
ONS—Brazilian Independent System Operator (2015) Annual report of streamflow forecasting 2015, viewed 10 January 2021. www.ons.org.br (**in portuguese**)
ONS—Brazilian Independent System Operator (2016) Annual report of streamflow forecasting 2016, viewed 10 January 2021. www.ons.org.br (**in portuguese**)
ONS—Brazilian Independent System Operator (2017) Annual report of streamflow forecasting 2017, viewed 10 January 2021. www.ons.org.br (**in portuguese**)
Polyak BT (1964) Some methods of speeding up the convergence of iteration methods. USSR Comput Math Math Phys 4:1–17
Pontes PRM, Fan FM, Fleischmann AS, Rodrigo CDP, Buarque DC, Siqueira VA, Jardim PF, Sorribas MV, Collischonn W (2017) MGB-IPH model for hydrological and hydraulic simulation of large floodplain river systems coupled with open source GIS. Environ Model Softw 94:1–20
Quan H, Srinivasan D, Khosravi A (2014) Short-term load and wind power forecasting using neural network-based prediction intervals. IEEE Trans Neural Netw Learn Syst 25:303–315
Rasmussen PF, Salas JD, Fagherazzi L, Rassam J, Bobée B (1996) Estimation and validation of contemporaneous PARMA model for streamflow simulation. Water Resour Res 32:3151–3160
REN21 (2018) Renewables 2018 Global Status Report
Rozante JR, Moreira DS, Goncalves LGG, Vila DA (2010) Combining TRMM and surface observations of precipitation: technique and validation over South America (2010) Weather and Forecasting 25:885–894
Ruder S (2017) An overview of gradient descent optimization algorithms. CoRR. arXiv:1609.04747v2.
Russom P (2011) Big data analytics. TDWI best practices report
Santos CAG, Silva GBL (2014) Daily streamflow forecasting using a wavelet transform and artificial neural network hybrid models. Hydrol Sci J 59(2):312–324
Schmidhuber J (2015) Deep learning in neural networks: an overview. Neural Netw 61:85–117
Sitterson J, Knightes C, Parmar R, Wolfe K, Muche M, Avant B (2017) An overview of rainfall-runoff model types. EPA—United States Environmental Protection Agency
Souza FC, Legey LFL (2008) Brazilian electricity market structure and risk management tools. IEEE PES General Meeting.
Souza SA, Costa FS, Xavier LNR, Maceira MEP, and Damázio JM (2010) PREVIVAZ—improving weekly streamflow time series forecasts with current hydrologic state of the river basin. International Workshop Advances in Statistical Hydrology
Sttari MT, Yurekli K, Pal M (2012) Performance evaluation of artificial neural network approaches in forecasting reservoir inflow. Appl Math Model 36:2649–2657
Taormina R, Chau K (2015) Data-driven input variable selection for rainfall-runoff modeling using binary-coded particle swarm optimization and extreme learning. J Hydrol 529:1617–1631
TensorFlow (2019) An open-source machine learning framework for everyone. www.tensorflow.org. Accessed 10 Jan 2021 (**in portuguese**)
Tian Y, Pei K, Jama S, and Ray B (2018) DeepTest: automated testing of deep-neural-network-driven autonomous cars. In: ICSE '18 proceedings of the 40th international conference on software engineering, pp 303–314
Tongal H, Booij MJ (2018) Simulation and forecasting of streamflows using machine learning models couple with base flow separation. J Hydrol 564:266–282
Wan L, Zeiler M, Zhang S, LeCun Y, Fergus R (2013) Regularization of neural networks using drop connect. In: International conference on machine learning (ICML)
Witten IH, Frank E, Hall MA, Pal CJ (2017) Data mining practical machine learning tools and techniques, 4th edn. Morgan Kaufmann, Burlington

Xu B, Wang N, Chen T, Li M (2015) Empirical evaluation of rectified activations in convolutional network. CoRR, abs/1505.00853

Yaseen ZM, El-shafie A, Jaafar O, Afan HA, Sayl KN (2015) Artificial intelligence based model for stream-flow forecasting: 2000–2015. J Hydrol 530:829–844

Yaseen ZM, Jaafar O, Deo RC, Kisi O, Adamowski J, Quilty J, El-Shafie A (2016) Stream-flow forecasting using extreme learning machines: a case study in a semi-arid region in Iraq. J Hydrol 542:603–614

Zadeh MR, Amin S, Khalili D, Singh VP (2010) Daily outflow prediction by multi layer perceptron with logistic sigmoid and tangent sigmoid activation functions. Water Resour Manag 24:2673–2688

Zealand CM, Burn DH, Simonovic SP (1999) Short term streamflow forecasting using artificial neural networks. J Hydrol 214:32–48