Neural Computing and Applications (2022) 34:13253–13266
https://doi.org/10.1007/s00521-021-06746-5

S.I.: DEEP LEARNING FOR TIME SERIES DATA

Hydropower production prediction using artificial neural networks: an Ecuadorian application case

Julio Barzola-Monteses[1,2] · Juan Gómez-Romero[1] · Mayken Espinoza-Andaluz[3] · Waldo Fajardo[1]

Received: 10 May 2021 / Accepted: 10 November 2021 / Published online: 15 December 2021

Abstract
Hydropower is among the most efficient technologies to produce renewable electrical energy. Hydropower systems present multiple advantages since they provide sustainable and controllable energy. However, hydropower plants' effectiveness is affected by multiple factors such as river/reservoir inflows, temperature, electricity price, among others. The mentioned factors make the prediction and recommendation of a station's operational output a difficult challenge. Therefore, reliable and accurate energy production forecasts are vital and of great importance for capacity planning, scheduling, and power systems operation. This research aims to develop and apply artificial neural network (ANN) models to predict hydroelectric production in Ecuador's short and medium term, considering historical data such as hydropower production and precipitations. For this purpose, two scenarios based on the prediction horizon have been considered, i.e., one-step and multi-step forecasted problems. Sixteen ANN structures based on multilayer perceptron (MLP), long short-term memory (LSTM), and sequence-to-sequence (seq2seq) LSTM were designed. More than 3000 models were configured, trained, and validated using a grid search algorithm based on hyperparameters. The results show that the MLP univariate and differentiated model of one-step scenario outperforms the other architectures analyzed in both scenarios. The obtained model can be an important tool for energy planning and decision-making for sustainable hydropower production.

Keywords Artificial neural network · Hydropower production forecasting · LSTM · MLP · Monthly electricity production · Sequence to sequence

1 Introduction

Hydropower is a renewable energy source where electrical energy is derived from water's potential energy moving from higher to lower elevations. Hydropower is a mature technology and widely used; in 2019, a total of 170 countries/territories in the world reported to have installed capacity and generated hydropower. Hydropower is among the most efficient technologies for producing renewable electrical energy, with a typical efficiency of 90%. Hydropower systems are cost-competitive: Today, it is the only renewable technology that produces electricity at an equal or lower cost, compared to thermal energy sources like coal, oil, or gas, typically in the range of USD 2–5c per kWh [1, 2].

Ecuador is an emerging hydropower actor in Latin America. The country is currently implementing the National Master Plan for Electrification 2016–2025 to cover more than 90% of the national electrical demand with hydroelectric sources [3]. According to the last official report of Ecuador's electric sector statistics in 2018, Ecuador's hydroelectric capacity (5036.43 MW) represented 62.58% of the total capacity (8048.11 MW). Generation from hydroelectric systems in 2018 amounted to 70.71% of the total production [4].

✉ Julio Barzola-Monteses
jbarzola@correo.ugr.es

[1] Department of Computer Science and Artificial Intelligence, Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación, Universidad de Granada, 1807 Granada, Spain

[2] Artificial Intelligence and Information Technology Research Group, University of Guayaquil, 090514 Guayaquil, Ecuador

[3] Centro de Energías Renovables y Alternativas, Facultad de Ingeniería Mecánica y Ciencias de la Producción, Escuela Superior Politécnica del Litoral, 09-01-5863, Guayaquil, Ecuador

Sustainability, controllable energy, and the capability of quickly responding to surges of demand in the grid are among the main advantages of hydroelectric plants. Reliability offered for hydroelectric plants is a pivotal characteristic that cannot be obtained by solar or wind sources due to intermittence. Notwithstanding the multiple advantages, hydropower plants' effectiveness is affected by factors such as river/reservoir inflows, temperature, seasonal demand, abrupt demands, gross domestic product, electricity price, and particularly their complicated correlations with meteorological and human phenomena. All these aspects make the prediction/recommendation of station operational output a difficult challenge [5]. At the same time, reliable and accurate energy production forecasts are vital to electric energy operators to improve the capacity planning, scheduling, and operation of power systems. Finally, depending on the needs of the electric operator, these forecasts can be accumulated on different time scales: short (hours or days ahead), medium (from a week to months ahead), or long term (years ahead) [6].

This research aims to explore and evaluate the capabilities of ANN models for predicting hydroelectric production in the short and medium terms. We focus our experiments on historical data from Ecuador, but the approach can be extended to any other country. Since hydropower strongly depends on meteorological conditions, rainfall data play a crucial role in our study. Previous works identified some limitations of statistical time analysis techniques [7, 8]: the difficulties in incorporating multiple exogenous variables and the lack of accuracy for long-term predictions. Following the success of neural networks in many areas [9, 10], the departing hypothesis of this paper is that ANN models can better capture the dynamics of hydroelectric production and, therefore, provide better results in our use case—even though the number of observations is not very large. In the literature, there are a few contributions studying hydroelectric production in countries such as China, Serbia, and Brazil [5, 11, 12]. Still, there are no similar studies in the literature applying recurrent neural networks and focused on a medium-size region during a large period in which the installed capacity has evolved, as is the case in Ecuador. The model obtained represents an interesting tool for energy planning and decision-making for sustainable hydropower production in Ecuador. Furthermore, the paper provides a methodology and tools for model training and configuration that can be extended to other contexts.

The rest of the paper is organized as follows: Sect. 2 summarizes the ANN techniques used and presents related works on the topic. The experimental design applied in the current study is detailed in Sect. 3. Experimental evaluations and discussion of the results based on the models are shown in Sect. 4. Finally, Sect. 5 includes concluding remarks and directions for future work.

2 Background

2.1 Neural networks for time series prediction

A time series is a set of data samples collected at regular time intervals. Time series analysis comprises a vast collection of techniques for analyzing historical temporal data to extract meaningful features or information. Time series forecasting using deep learning (a family of machine learning techniques based on artificial neural networks) has shown higher accuracy than other traditional techniques [13].

ANN is a nonlinear computational model loosely inspired by the human brain. In general, an ANN is made up of several layers of neurons connected through weighted links. The term "deep learning" means that there are many of these layers and neurons. The output of the network is calculated as the composition of the calculations performed by each neuron. Training an ANN means adjusting the weights to approximate the (unknown) function that relates the (known) inputs and outputs. There are commonly three aspects that characterize an ANN: (i) the interconnection pattern between the neurons of the different layers; (ii) the optimization algorithm to learn the weights of the interconnections, and (iii) the activation function that converts a neuron's weighted input to its output activation [14].

We consider a feed-forward neural network (FFNN) and two recurrent neural networks (RNN) in this work. Specifically, we use multilayer perceptron (MLP), Long Short-Term Memory (LSTM) and sequence-to-sequence LSTM (seq2seq LSTM) models, which are briefly presented below.

2.1.1 Multilayer perceptron model (MLP)

Feed-forward neural networks (FFNNs) are a type of ANN in which the computations proceed only in the forward direction. The most common FFNN is the multilayer perceptron (MLP), in which each neuron in a layer is connected to every neuron in the next layer. MLP networks are typically trained with the stochastic gradient descent method, which iteratively updates the network's weights according to the error between the obtained and the expected outputs. This error flows backward through the network to calculate the gradient at each neuron, a process called backpropagation [15]. Despite its simplicity, MLP has achieved updated performance in various supervised and unsupervised machine learning applications. Its success highly depends on the independence assumption

among the training and test data [16]. In this work, we use MLPs to process subsequences of fixed length, i.e., the size of the input layer of the MLP determines the length of the sample of the sequence.

2.1.2 Long short-term memory model (LSTM)

Recurrent Neural Networks (RNNs) are ANN types that allow cycles in the network, i.e., computations can proceed from one layer to a previous one. In this way, RNNs use the current inputs and previous calculations to compute the network output, thus keeping track of an internal state. RNNs are trained with the backpropagation through time (BPTT) method [15], which applies backpropagation to an "unrolled" version of the network—unrolling means obtaining an equivalent acyclic version of the network by replicating the components involved in the recurrence and sharing their weights.

Long short-term models (LSTMs) are a particular type of RNNs purposely designed to learn both short- and long-term temporal dependencies in time series data. Hochreiter and Schmidhuber designed this new architecture based on the concept of "emory blocks or gate units" in each hidden layer. They had as the primary motivation to solve the vanishing and exploding gradients problem [17–19]. Figure 1a shows a schematic diagram of an LSTM unit.

Given an input time series $x = \{x_1, x_2, \ldots, x_T\}$, the LSTM maps the input time series to two output time sequences $h = \{h_1, h_2, \ldots, h_T\}$ and $y = \{y_1, y_2, \ldots, y_T\}$ iteratively by updating the states of memory cells with the following procedure [16, 20]:

Each gate is a sigmoid unit that changes every element in [0, 1], i.e., they use a logistic function defined in Eq. (1):

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

Input gate i_t controls the input of new information that is going to be stored in the new cell state, which derives the following:

$$i_t = \sigma(w_{xi}x_t + w_{hi}h_{t-1} + b_i). \tag{2}$$

Forgetting gate f_t decides what information must be discarded from cell state, where:

$$f_t = \sigma\left(w_{xf}x_t + w_{hf}h_{t-1} + b_f\right). \tag{3}$$

Output gate o_t controls and filters the output information flowing out of the cell, where:

$$o_t = \sigma(w_{xo}x_t + w_{ho}h_{t-1} + b_o). \tag{4}$$

At each time t, the input features are computed by input x_t and the previous hidden state h_{t-1} by using the tanh function, as follows:

$$g_t = \tanh(w_{xc}x_t + w_{hc}h_{t-1} + b_c). \tag{5}$$

The memory cell is updated by moderated input features and the partial forgetting of the previous memory cell, which yields:

$$c_t = f_t * c_{t-1} + i_t * g_t. \tag{6}$$

Then, the hidden output state h_t is calculated by the output gate o_t and the memory cell c_t, as follows:

$$h_t = o_t * \tanh(c_t). \tag{7}$$

Finally, the output of LSTM y_t is computed by using the following expression:

$$y_t = \sigma\left(w_{hy}h_t + b_y\right). \tag{8}$$

In Eqs. (2)–(8), the matrices w_{xi}, w_{xf}, w_{xo}, and w_{xc} are the appropriate input weighting matrices, while w_{hi}, w_{hf}, w_{ho}, and w_{hc} are the recurrent weighting matrices; and w_{hy} represents the hidden output weight matrix. The vectors b_i, b_f, b_o, b_c, and b_y are the corresponding bias vectors.

LSTMs are trained with BPTT, considering that w_{hi}, w_{hf}, w_{ho}, and w_{hc} are the recurrent weighting matrices. They support the backpropagation of the gradient many time steps into the past, thus preventing the vanishing and exploding gradient problems. For a comprehensive description of the gradient computation of each component of the LSTM gate, we refer the reader to [21].

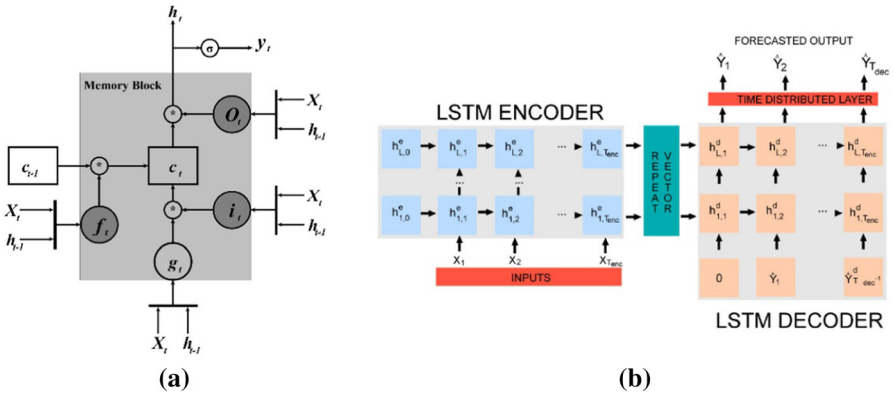

Fig. 1 Schematic diagrams: **a** LSTM unit and **b** encoder and decoder in the seq2seq model with L stacked layers

2.1.3 Sequence-to-sequence model (seq2seq)

The sequence-to-sequence model (seq2seq) is a neural architecture that translates one data sequence into another. It has a structure made up of two sub-modules: the encoder to read and encode the input sequence, and the decoder in charge of reading the encoded input sequence and generating the output sequence prediction. Seq2seq models are related to autoencoders, an unsupervised learning architecture aimed at regenerating the input from the output. While autoencoders use the same sequence as input and output, seq2seq allows supervised learning by accepting different sequences at both ends.

Seq2seq model was first introduced by Sutskever et al. in 2014 [22]. Each sub-model can be composed of recurrent neural networks such as gated recurrent unit (GRU) or LSTM [23] and trained accordingly. These models can address challenging sequence-to-sequence prediction problems. In this work, seq2seq LSTM to perform medium-term hydropower production forecasting is considered.

Figure 1b shows a schematic diagram of seq2seq LSTM. The encoded input sequence representation is repeated multiple times, once each step in the output sequence. This sequence of vectors is presented to the LSTM decoder. Simultaneously, the decoder output sequence is wrapped through the time distributed layer, which allows the wrapped layers to be used for each time step from the decoder [24].

Given an input time series $\chi_{1:T_{\text{enc}}} = \chi_1, \chi_2, \ldots, \chi_{T_{\text{enc}}}$, the encoder maps the input sequence time series to the fixed-length representation $h^e_{1:L,T_{\text{enc}}}$ [25]:

$$h^e_{1:L,T_{\text{enc}}} = f_{\text{enc}}\left(\chi_{1:T_{\text{enc}}}\right) \tag{9}$$

where $h^e_{1:L,T_{\text{enc}}}$ is the encoder's hidden and/or cell states for all stacked layers at time T_{enc}. Then, the decoder output $\hat{y}_{1:T_{\text{dec}}}$ is defined as follows:

$$\hat{y}_{1:T_{\text{dec}}} = f_{\text{dec}}\left(h^e_{1:L,T_{\text{enc}}}\right) \tag{10}$$

which is a sequence of T_{dec} predictions. A simplified way to denote the composition of the encoder and decoder is using the following expression:

$$\hat{y}_{1:T_{\text{dec}}} = f_{\text{enc}\rightarrow\text{dec}}\left(\chi_{1:T_{\text{enc}}}\right). \tag{11}$$

2.2 Related works

Before mentioning some related works, it is necessary to know the classification of hydroelectric power systems. Hydroelectric projects are mainly classified as run of river (RoR) hydropower plant, storage (reservoir) hydropower plant (SHP), pumped-storage hydropower (PSH), and in-stream technologies (Hydrokinetic). An RoR hydropower is a plant where little or no water storage is provided; it generates electricity from the river's available flow. An SHP includes a dam and a reservoir to impound water, store and release later when needed. In a PSH plant, water is pumped from a lower reservoir into an upper reservoir when electricity generation exceeds demand and is released back from the upper reservoir through turbines to generate electricity when demand exceeds the supply. Finally, a hydrokinetic plant can be derived from water's movement (kinetic energy) in rivers, streams, canals, tidal flow, and ocean currents [2].

Table 1 shows the main works related to hydroelectric production's prediction in the last five years. There is more than one approach to develop forecasts, such as (i) statistical techniques to derive mathematical relationships between dependent variables and independent variables, (ii) hydrological modeling for the characterization of real hydrological systems using physical models and computational simulation, (iii) satellite mapping techniques in which high-resolution satellite images are considered in studies such as orography and hydrographic basins, (iv) optimization algorithms which are procedures that are executed iteratively comparing several solutions until an optimal solution is found, and (v) machine learning techniques which use computational methods to learn information directly from data without relying on a predetermined equation as a model.

The most related works to ours are [5, 11, 12], which also use ANNs for hydrologic (power) time series prediction. In [5], the authors present DeepHydro, a deep learning framework for multivariate time series prediction based on latent recurrent neural networks. The experimentation focused on a 2-year dataset of the stations located across the Dadu River (China) and hourly forecasts. The main influencing variables considered were temperature and water flow, while we use precipitations—easier to obtain—as a proxy for the latter. In contrast, we consider larger periods and several watersheds, and more importantly, a dynamic context—Ecuador's installed capacity varies during the study period. A more comprehensive comparison of DeepHydro and fine-tuned LSTMs remains as future work.

In [11], the hydrological flow in Southwest Serbia was estimated instead of the production. Precipitations were identified as a critical predictor of flow, which is a fundamental assumption of our work—afterward validated in the experimentation. Our methodology for hyperparameter optimization is inspired by their approach, although we apply a more comprehensive set of techniques (i.e., MLPs with different numbers of input nodes, LSTM, seq2seq) beyond the MLPs used in their paper.

Table 1 Selected related works of hydropower production forecasting

References	Geographic location	Hydropower projects	Regressive variable	Prediction variable	Approach to developing forecasts
Zhou et al. [5]	Hydropower stations on the Dadu River (China)	Storage (reservoir) based	Water flow External factors (meteorology, time, and sale price)	Power production	DeepHydro—recurrent neural networks
Kostić et al. [11]	southwestern Serbia	Storage (reservoir) based	Air temperature Precipitation	Hydrological flow rate Power production	Multilayer feed-forward perceptron (MLP)
Lopes et al. [12]	Amazon region (Brazil)	run-of-river	Precipitation	Monthly potential hydropower generation	Group method of data handling MLP
Jung et al. [26]	Han River basin (South Korea)	Storage (reservoir) based	Precipitation Humidity Temperature Wind speed	Runoff prediction	Multilayer feed-forward perceptron (MLP)
Razi et al. [27]	Peninsular Malaysia	run-of-river	Net head Water flow rate	Power production	Statistical analysis is applied
Oyerinde et al. [28]	Niger Basin (West Africa)	Storage (reservoir) based	Precipitation Evapotranspiration	Power production	A hydrological model is applied
Dehghani et al. [29]	Dez dam (Iran)	Storage (reservoir) based	Precipitation Water flow rate Power Production	Power production	Grey wolf optimization Adaptive neuro-fuzzy inference system (ANFIS)
Tamm et al. [30]	North Estonia	Run-of-river	Digital elevation map Land use map Soil map Weather data	Hydropower potential	Soil and Water Assessment Tool (SWAT) model
Chen and Zhong [31]	Tankeng hydropower station (China)	Storage (reservoir) based	Reservoir inflow Electricity price Hydropower consumption rate	Power production	Multi-time-scale coupling operation Model dynamic Bayesian network Model probability-based prediction model
Contreras et al. [32]	Poqueira River basin (South of Spain)	run-of-river	Climate service Historical local data Seasonal forecast of water inflow	Power PRODUCTION	Quantile mapping method
Hidalgo et al. [33]	São Francisco river basin (Brazil)	Storage (reservoir) based	Air temperature Precipitation	Power production	Hydrological-hydropower model
Farfán et al. [34]	Machángara sub-basin of the Paute river basin (Ecuador)	Storage (reservoir) based	Meteorological data	Water flow	WEAP and GR2M physical models ANN models ANN hybrid model

Similarly, the research work in [12] focused on MLPs and the group method of data handling (GMDH)—a variation in MLP that allows zeroing selected nodes of the network—to predict hydrological production in the Amazon (Brazil). They also used data from several years and rainfall data (aggregated from several basins) as the exogenous variable. Another study developed in South Korea performs a similar analysis considering climate change scenarios and using MLPs [26]. Again, our paper considers more recent techniques aimed explicitly at time series processing, and the comparison with GMDH can be a prospective direction for future work.

3 Experimental setup

3.1 Data

Data for the current study, corresponding to the 2000–2015 year period, have been obtained from official and governmental institutions of Ecuador, namely the Electricity Regulation and Control Agency (ARCONEL) [35] and the National Institute of Meteorology and Hydrology (INAMHI) [36]. The dataset is built considering the data collected from the annual statistical reports of both institutions. The resolution of the collected dataset is 1 month.

Figure 2 depicts the monthly gross production (MGP) of hydroelectric systems (GWh) and the total average monthly precipitation (mm) of the three main hydrographic basins of Ecuador [8]. The entire dataset is divided into a train and test set with zero overlaps. The first 75% of all data were assigned as the training set and the last 25% as the test set.

A time series can be considered stationary if the mean and variance are constant and there are no significant trends and seasonal variations. The MGP series does not present stationary characteristics due to a significant trend. Then, the first difference is applied to decrease the trend, i.e., the series is detrended [37]. Both series were used in the proposed models: original MGP series, i.e., without differentiation, and MGP series with differentiation (D).

3.2 Model features and targets

The experimental design considers a univariate and bivariate time series problem. For the univariate case, only one variable or feature is considered as input (regressor variable) and output (predictor variable) of the model (black box). In this univariate case, the hydroelectric production variable is considered. On the other hand, for the bivariate case, two variables are considered, i.e., two regressor variables at the black box's input and one at the output as a prediction variable. In this bivariate case, hydroelectric production and precipitation are regressive variables, and hydroelectric production is kept as a predictor variable.

The training addressed in the models is of the supervised type. The generated samples to be passed to the model at the input and output have the following structure:

Let one input sample be represented as a matrix, $X \in \mathbb{R}^{T \times f}$, where T is the number of time steps and f is the number of features. As mentioned, the number of features was $f = 1$ (univariate case) and $f = 2$ (bivariate case), where each row of the matrix X is defined as follows:

$$\text{Univariate case} : x_{[t]} = \left[\text{MGP}_{[t]}\right] \tag{12}$$

$$\text{Bivariate case} : x_{[t]} = \left[\text{MGP}_{[t]} \quad \text{Prec}_{[t]}\right] \tag{13}$$

where $t \in 1, \ldots, T$. For each input sample, one target sample is generated, represented by a vector $y \in \mathbb{R}^{N \times 1}$, where N represents the number of predicted time steps from T. The target vector represents the actual MGP vector, which is given by

$$y = \left[\text{MGP}_{[1]}, \ldots, \ \text{MGP}_{[n]}, \ldots, \text{MGP}_{[N]}\right] \tag{14}$$

where $y_{[n]}$ is the actual MGP value at time step n, and $n \in 1, \ldots, N$. This targeting vector is used to compare with the predicted MGP vector, $\hat{y} \in \mathbb{R}^{N \times 1}$.

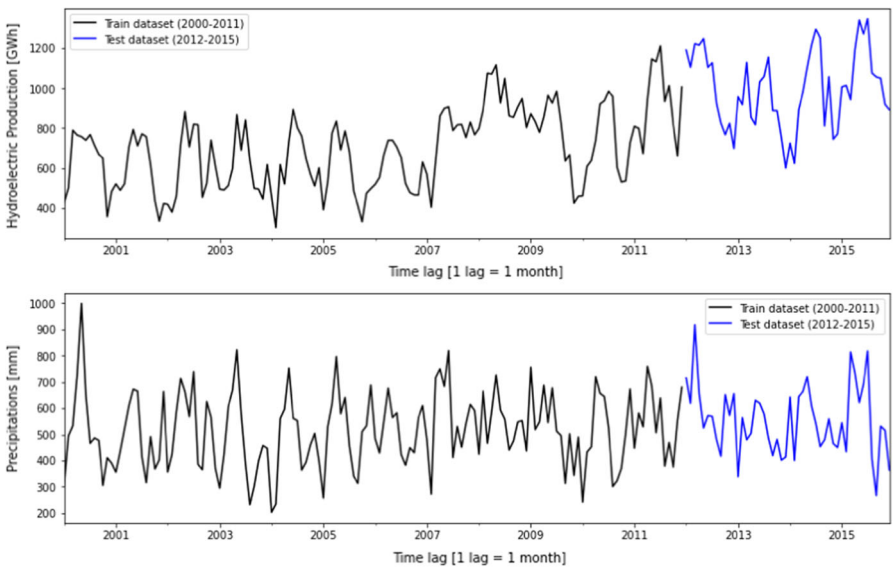

Fig. 2 Monthly hydroelectric production (up) and total precipitation (down) of the three considered watersheds during 2000–2015

3.2.1 Problem framing

A forecast problem that requires a prediction of the next time step is named one-step forecast model. On the other hand, a forecast problem involving a prediction of more than a one-time step is the multi-step forecast model. The current work considers both forecast problems as scenarios. The one-step forecast model implies a month of prediction, while the multi-step forecast model implies twelve months, i.e., one year. The two mentioned scenarios are presented as follows:

One-step scenario:

- Univariate case: Given a recent hydroelectric production, what is the expected hydroelectric production for one step ahead?
- Bivariate case: Given recent hydroelectric production and precipitations, what is the expected hydroelectric production for one step ahead?

Multi-step scenario:

- Univariate case: Given a recent hydroelectric production, what is the expected hydroelectric production for twelve steps ahead?
- Bivariate case: Given a recent hydroelectric production and precipitations, what is the expected hydroelectric production for twelve steps ahead?

3.2.2 Data transformation

Normalization is applied to each variable's data set and transformed into values between -1 and $+1$. Input and output are rescaled from one range of values (original values) to a new range of values. The rescaling is often accomplished by using a linear interpretation formula such as

$$x' = \frac{(x_i - x_{\min})}{(x_{\max} - x_{\min})}(x_{\text{new\_max}} - x_{\text{new\_min}}) + x_{\text{new\_min}} \tag{15}$$

where x' is the normalized value, x_i is the real value, $x_{\max}$ and $x_{\min}$ are the maximum and minimum values of the variable, respectively. In this work, $x_{\text{new\_max}}$ and $x_{\text{new\_min}}$ are $+1$ and -1, respectively. Equation (15) is named Min–Max normalization, which can preserve all data relationships [38].

3.3 Hyperparameters

Hyperparameters are configuration parameters external to the model itself, whose values generally cannot be estimated from the training data set and are specified by the designer to adjust the learning algorithms. There are mainly three types of hyperparameters: (i) structure and topologies, such as number of layers, number of neurons, their activation functions, and others; (ii) optimization, such as epochs, batch size, learning rate, momentum, and similar; and (iii) Regularization, such as dropout probability, L2 regularization coefficient, and others [25, 39].

In the current work, more than 3000 models were trained. Considering a grid search algorithm, a tuning process was applied to the implemented models by varying hyperparameters to find the configurations that yield the best generalization on the validation data set. The grid search procedure exhaustively considered all the hyperparameter combinations and selected the best subset among them. This has a high computational cost that can be too expensive for models and datasets larger than ours. Hyperparameter optimization algorithms such as HyperBand [40] and Bayesian Optimization [41] can be applied in these cases.

Accordingly, MLP, LSTM, and seq2seq LSTM architectures for the two scenarios specified above have been configured. For the case of the MLP and LSTM models, configurations with one hidden layer are considered. Table 2 shows the variation in hyperparameters used in each of the implemented models.

In Table 2, n_input is the number of prior inputs to use as input for the model, n_nodes is the number of nodes/units to use in the hidden layer, n_epochs is the number of training epochs, n_batch is the number of samples to include in each mini-batch, act_hid is the activation function to use in the hidden layer, and act_out is the activation function to use in the output layer. For the case of seq2seq LSTM, nodes_enc and nodes_dec are the number of LSTM units to use in the encoder and decoder, respectively; nodes_dense is the number of nodes to use in the fully connected layer; act_enc and act_dec are the activation functions used in the encoder and decoder, respectively.

Table 3 shows all structures used in different configurations with hyperparameters during the execution of the experiments. For simplification reasons, each model is represented through its corresponding acronyms.

On the other hand, due to neural networks' stochastic nature, each structure configuration is trained and validated ten times during the hyperparameter tuning process. The mean of the RMSE errors is determined and registered.

3.4 Performance evaluation

3.4.1 Model evaluation

As known, k-fold cross-validation does not work for time series data because they ignore its temporal relation. Therefore, a rolling-forecast method will be used, also

Table 2 hyperparameters considered for experiment models MLP, LSTM and seq2seq LSTM

MLP and LSTM models		seq2seq LSTM model	
Hyperparameter	Values	Hyperparameter	Values
n_input	Range: 1 to 24, incrementing by powers of 2	n_input	Range: 1 to 24, incrementing by powers of 2
n_nodes	Range: 2 to 64, incrementing by powers of 2	nodes_enc	Range: 2 to 64, incrementing by powers of 2
n_epochs	Range: 25 to 150, incrementing by powers of 25	nodes_dec	Range: 2 to 64, incrementing by powers of 2
n_batch	Range: 2 to 16, incrementing by powers of 2	nodes_dense	Range: 2 to 64, incrementing by powers of 2
act_hid	'Sigmoid', 'tanh', 'relu', 'linear'	n_epochs	Range: 25 to 150, incrementing by powers of 25
act_out	'Sigmoid', 'tanh', 'relu', 'linear'	n_batch	Range: 2 to 16, incrementing by powers of 2
		act_enc	'Sigmoid', 'tanh', 'relu', 'linear'
		act_dec	'Sigmoid', 'tanh', 'relu', 'linear'

Table 3 Acronyms of the ANN models considered during the execution of the experiments

Scenario	Time series problem	Model	Acronyms
One-step	Univariate	Multilayer perceptron without differentiation—univariate	MLP-uni
	Univariate	Multilayer PERCEPTRON with differentiation—univariate	MLP-D-uni
	Univariate	Long short-term memory without differentiation—univariate	LSTM-uni
	Univariate	Long short-term memory with differentiation—univariate	LSTM-D-uni
	Bivariate	Long short-term memory without differentiation—bivariate	LSTM-bi
	Bivariate	Long short-term memory with differentiation—bivariate	LSTM-D-bi
Multi-step	Univariate	Multilayer perceptron without differentiation—univariate	MLP-uni
	Univariate	Multilayer perceptron with differentiation—univariate	MLP-D-uni
	Univariate	Long short-term memory without differentiation—univariate	LSTM-uni
	Univariate	Long short-term memory with differentiation—univariate	LSTM-D-uni
	Univariate	Sequence-to-sequence LSTM without differentiation—univariate	S2S-uni
	Univariate	Sequence-to-sequence LSTM with differentiation—univariate	S2S-D-uni
	Bivariate	Long short-term memory without differentiation—bivariate	LSTM-bi
	Bivariate	Long short-term memory with differentiation—bivariate	LSTM-D-bi
	Bivariate	Sequence-to-sequence LSTM without differentiation—bivariate	S2S-bi
	Bivariate	Sequence-to-sequence LSTM with differentiation—bivariate	S2S-D-bi

called walk-forward model (WFM) validation. WFM partitions data before and after a selected time point and uses each partition for training and validation, respectively. Since we are interested in predicting for the following years, this method better reflects the expected accuracy of the model in real use. Following [42], each time step of the test dataset is executed one at a time.

3.4.2 Performance evaluation

In literature reviews, in which model adjustments with energy observations are handled, the RMSE metric is the most used evaluation parameter [42]. Therefore, the RMSE is considered the main metric to select the best configuration model with the lowest forecast error. Others supporting metrics are the MAE and MAPE [20, 43–45].

4 Experimental results and discussion

In this section, the experimental results achieved with MLP, LSTM, and seq2seq LSTM models are presented and discussed. All experiments executed in this research were run on a machine with Intel Core i7-6500U CPU @2.50 GHz × 4, equipped with 16 GB physical memory with running operating system Ubuntu 20.04 v. All configuration models were implemented in Python 3.8, using the TensorFlow library.

4.1 One-step scenario

After carried out the experiments and fine-tuning hyperparameters using the train dataset and WFM validation on the test dataset, the best topology neural network achieved is as follows: n_input = 1, n_nodes = 64 neuron, n_epochs = 25, n_batch = 1, act_hid = sigmoid and act_out = sigmoid. Table 4 shows the top 10 models based on the RMSE metric.

For this scenario, six structures shown in Table 3 were tested; however, only one appears in the top 10. In this case, MLP structures outperform both LSTM univariate and bivariate models. The best results were obtained using MLP with differentiation–univariate, i.e., applying differentiation to the data of the hydroelectric production variable resulted in better accurate predictions.

From Table 4, it can be noted that in the scenario of predicting a month from past months, the best-obtained models require a look back of one month. In addition, the following characteristics are observed: The number of nodes varies between 16 and 64. The epochs' values are 25 and 100. The act_hid can be sigmoid and linear functions, whereas for act_out, the model has better accuracy with sigmoid and tanh functions. Finally, the common hyperparameters are n_input = 1 and n_batch = 1.

Table 5 shows the accuracy results RMSE of the best models of each structure. For the MLP case, a better fit of the prediction data is achieved by transforming the series, i.e., differentiation is applied. Also, LSTM univariate and bivariate structures, the best metrics were obtained by differentiation.

From Table 5, it can be noted that MLP outperforms LSTM. For the case of LSTM structures, accuracy can improve if there is more than one regressor variable. For our case study, LSTM-D-bi forecasts improved about 38% compared to the best LSTM univariate. In this work and case study, MLP achieved better error metrics than RNN techniques for one-step scenarios.

LSTM models require more computational time to run. Figure 3 shows that the better average time is around 0.5 min per model and it is achieved with the univariate and differentiated MLP. Moreover, its RMSE metric is the lowest among all the one-step architectures. The univariate LSTM model has the worst time, 2.48 min/model. On the other hand, it is noted that when applying differentiation for the bivariate LSTM case, time improves substantially compared to LSTM-D-uni (about 15%). Based on the results, MLP-D-uni is the model with the best RMSE metric and time to run in this scenario.

4.2 Multi-step scenario

After carried out the experiments and fine-tuning hyperparameters using the train dataset and WFM validation in test dataset, the best topology neural network achieved is as follows: n_input = 5, n_nodes = 84 neurons, n_epochs = 100, n_batch = 16, act_hid = tanh and act_out = linear.

Table 6 shows the top 10 models based on the RMSE metric. For this scenario, ten structures shown in Table 3 were tested; however, only one type appears in the top 10. In this case, MLP outperforms both LSTM and Seq2seq LSTM models. The best model was MLP with differentiation—univariate. Moreover, to all the top ten models showed in Table 6, differentiation to the MGP series was applied, and better accurate predictions were achieved.

It is also noted that in this scenario of predicting twelve months from past months, the best-obtained models require a look back of 5 months, i.e., in this multi-step scenario, a lower amount of historical data is required. The following characteristics are also observed: The number of nodes varies between 84 and 128 for MLP-D. The epochs' values vary between 50 and 150. Finally, the common hyperparameters are n_input = 5, n_batch = 16, act_hid = tanh and act_out = linear.

The results achieved with LSTM are not so distant from those achieved by MLP. The RMSE difference between the best MLP-D and LSTM-D-uni is 12.2 GWh, i.e., about

Table 4 The top ten results from our hyperparameter search and WFM validation

Model	n_input	n_nodes	n_epochs	n_batch	act_hid	act_out	RMSE (GWh)
MLP-D-uni	1	64	25	1	Sigmoid	Sigmoid	4.09
MLP-D-uni	1	32	25	1	Sigmoid	Sigmoid	4.23
MLP-D-uni	1	64	25	1	Linear	tanh	4.60
MLP-D-uni	1	64	50	1	Linear	tanh	4.69
MLP-D-uni	1	16	50	1	Sigmoid	Sigmoid	4.78
MLP-D-uni	1	32	25	1	Linear	tanh	4.95
MLP-D-uni	1	64	75	1	Linear	tanh	5.04
MLP-D-uni	1	32	50	1	Linear	tanh	5.23
MLP-D-uni	1	64	100	1	Linear	tanh	5.28
MLP-D-uni	1	16	25	1	Sigmoid	Sigmoid	5.29

Table 5 Accuracy of RMSE results for all models and all cases of one-step scenarios

MLP		LSTM			
MLP-uni	MLP-D-uni	LSTM-UNI	LSTM-D-UNI	LSTM-bi	LSTM-D-BI
195.1	**4.09**	177.68	15.49	170.41	**11.23**

The number is the average on runs. The models with the best RMSE metrics are in bold

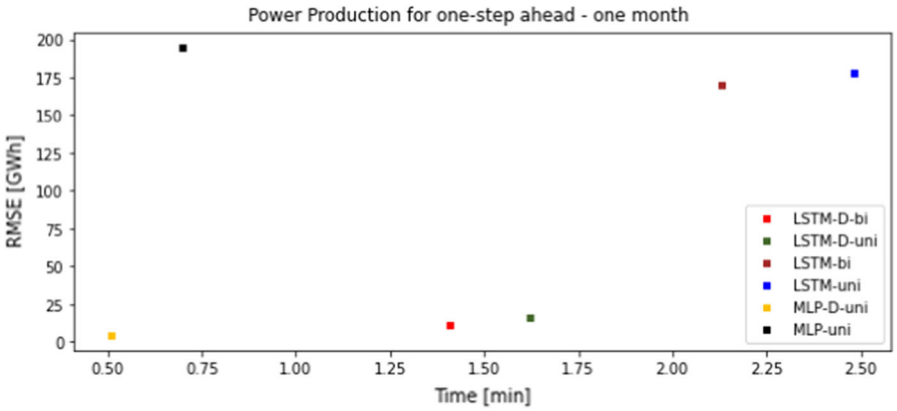

Fig. 3 Execution times of models contrasted with the RMSE of the best configurations. Results are obtained from the six architectures analyzed in this scenario

Table 6 The top ten results from the hyperparameter search (average of 12 steps ahead)

Model	n_input	n_nodes	n_epochs	n_batch	act_hid	act_out	RMSE (GWh)
MLP-D-uni	5	84	100	16	tanh	Linear	123.84
MLP-D-uni	5	84	75	16	tanh	Linear	123.84
MLP-D-uni	5	128	150	16	tanh	Linear	123.93
MLP-D-uni	5	94	75	16	tanh	Linear	124.00
MLP-D-uni	5	128	50	16	tanh	Linear	124.09
MLP-D-uni	5	94	125	16	tanh	Linear	124.12
MLP-D-uni	5	84	75	16	tanh	Linear	124.16
MLP-D-uni	5	94	50	16	tanh	Linear	124.18
MLP-D-uni	5	88	75	16	tanh	Linear	124.18
MLP-D-uni	5	104	125	16	tanh	Linear	124.19

9.9%. The LSTM-D-uni configuration ranks second with the best accuracy among the ten multi-step architectures considered in Table 3. Despite the limited number of observations, LSTM univariate architecture outperforms LSTM and seq2seq bivariate models. This result should be considered for future studies applied to the energy area.

Table 7 shows the accuracy results RMSE of the best model of each architecture. In general, the best RMSE values in each architecture considered in this multi-step scenario have been achieved by applying differentiation to the MGP series. Therefore, during data preprocessing, the MGP series differentiation improves the 12-month sequence of hydroelectric production prediction error.

Interestingly enough, there are no significant differences in the errors for each one of the 12-step predictions, particularly for the MLP. Figure 4 shows the validation errors of each month for the best MLP, LSTM, and seq2seq model highlighted in Table 7. These are average errors

Table 7 Accuracy of RMSE results for all models and all cases of multi-step scenarios

MLP		LSTM				Seq2Seq			
MLP-uni	MLP-D-uni	LSTM-UNI	LSTM-D-UNI	LSTM-bi	LSTM-D-BI	S2S-uni	S2S-D-uni	S2S-bi	S2S-D-bi
154.35	**123.84**	173.56	**136.04**	190.68	162.01	193.12	**140.42**	179	143.53

The number is the average of 12 steps ahead and multiple runs. The models with the best RMSE metrics are in bold

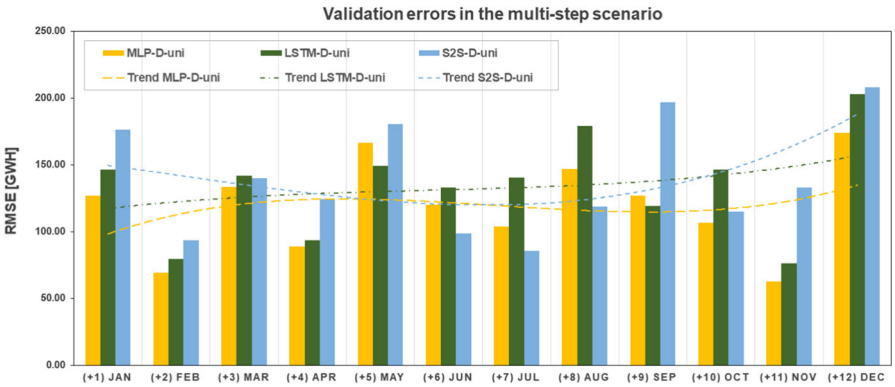

Fig. 4 Validation errors in the multi-step scenario (validation set)

since the validation data contain four years (2012–2015). As expected, we can find a slight trend in all of them to have larger errors in the farthest predictions, as indicated by the polynomial regression lines (dashed, polynomial degree = 3). MLP and LSTM are better in the shorter (1–4 months) and the longer terms (11–12 months), while seq2seq behaves well in the medium term (6–8 months). This opens the possibility to build an ensemble method, although, given the size of our dataset, it could lead to overfitting.

Figure 5 shows that the best average times per model are achieved with MLP architectures. Nevertheless, the best error between both structures is the MLP-D, with an execution time of around 0.18 min. It is noted that the S2S-uni model has the worst time, 2.95 min; it is also among the worst RMSE errors in this scenario. On the other hand, LSTM structures require a longer execution time than the MLP structure. Although LSTM-D-uni occupies the second place with the best metric among the ten structures analyzed, its runtime is higher than MPL-D by a factor of 4.

4.3 Scenario evaluation

As mentioned in the experiment setup section, the results' randomness is manifested by the neural networks' stochastic nature. Figures 6 and 7 show the box diagrams of hydropower production predictions' RMSE, MAE, and MAPE errors. The mentioned errors are obtained from 30 repetitions of each best architecture of MLP, LSTM, and sequence to sequence in each scenario analyzed.

Figure 6 depicts the error distribution of one-step scenarios; in this visual comparison of box plots, MLP-D-uni outperforming the other two structures LSTM is ratified. On the other hand, Fig. 7 shows the error distribution found when the multi-step scenario is analyzed. In this case, it is again confirmed that MLP-D-uni outperforms the other structures LSTM and seq2seq LSTM.

In this work, a one-step forecast outperforms the multi-step forecast scenario. Although MLP is the best ANN model for predicting power production, RNN bivariate one-step ranks second among all analyzed structures. In contrast, MLP-uni one-step ranks last among the sixteen structures analyzed. This result may be a consequence of training the model without differentiation MGP series. Finally, Fig. 7 shows that the detrended S2S-D-uni outperforms LSTM and seq2seq LSTM, both bivariate. Despite the limited number of dataset observations, this sequence-to-sequence model ranks sixth among all the structures analyzed.

ANN presents better forecast error results compared to ARIMA and ARIMAX statistical techniques. Table 8 and Fig. 8 show the comparative forecast errors for 2015

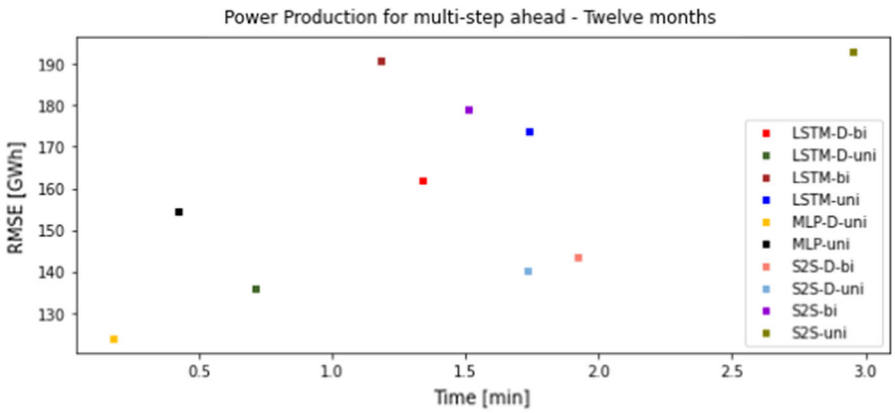

Fig. 5 Execution times of models contrasted with the RMSE of the best configurations. Results were obtained from 10 architectures analyzed in this scenario

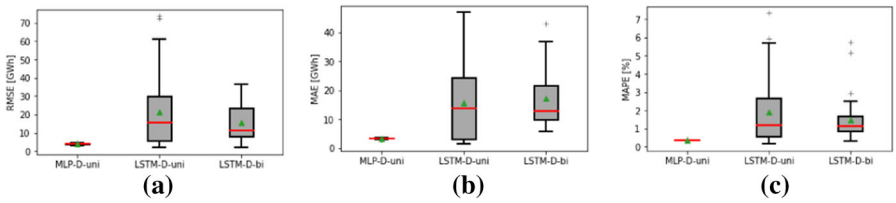

Fig. 6 Box plot errors of the experiments carried out with the best MLP and LSTM models in the one-step scenario. **a** RMSE, **b** MAE, and **c** MAPE. The red line represents the median, while the green triangle corresponds to the mean

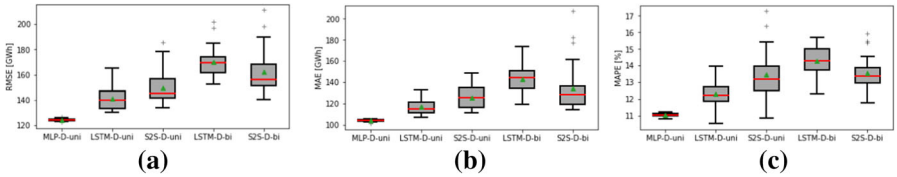

Fig. 7 Box plot errors of the experiments carried out with the best MLP, LSTM, and seq2seq LSTM models in the multi-step scenario. **a** RMSE, **b** MAE, and **c** MAPE. The red line represents the median, while the green triangle corresponds to the mean

Table 8 Forecast errors of each technique—2015

Technique	RMSE	MAE	MAPE
ARIMA [7]	193.04	161.65	14.32
ARIMAX [8]	111.54	86.52	8.44
ANN one-step scenario (MLP-D-uni)	4.11	3.39	0.32
ANN multi-step scenario (MLP-D-uni)	110.08	99.96	8.95

obtained with ARIMA and ARIMAX in previous studies [7, 8] that have considered the same dataset and variables.

5 Conclusion and future work

In this study, multiple ANNs were applied to forecast the hydroelectric production of Ecuador. Experiments using a dataset of MGP of hydroelectric systems (GWh) and total average monthly precipitation (mm) of the three main hydrographic basins of Ecuador were carried out.

According to the forecast horizon, two scenarios were considered: one-step (1 month) and multi-step (12 months). More than 3000 MLP, LSTM, and seq2seq LSTM models were configured, trained, and validated. RMSE, MAE, and MAPE were used to select the best model for each type of architecture. With limited observations, MLP with differentiation, univariate and bivariate series obtained the best results. These results evidence that ANN models surpass the traditional statistical models applied in time series. Also, they suggest that RNN models, such as LSTM and seq2seq, are not essentially superior in this problem setup to MLPs when hyperparameters are appropriately selected.

On the other hand, the average execution times of the models in one-step and multi-step scenarios are 1.48 and

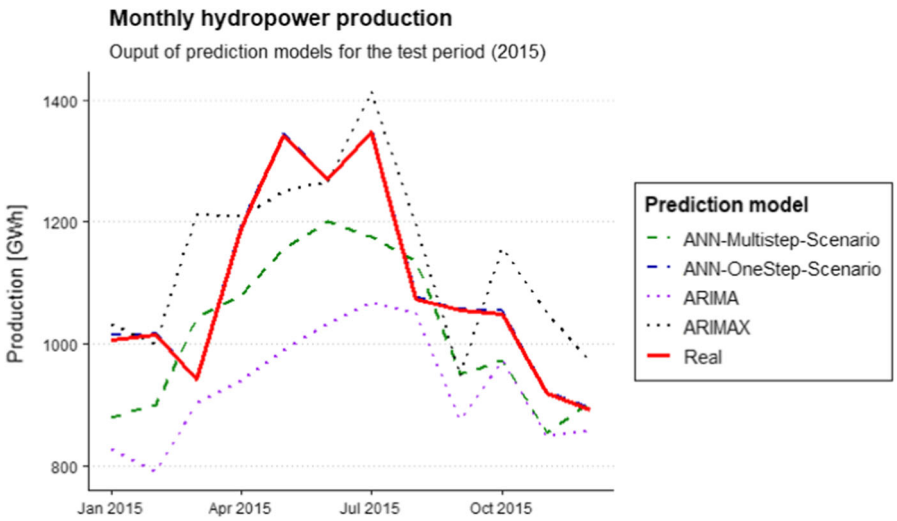

Fig. 8 Comparison between actual and predicted values 2015 with ANN (MLP-D-uni) and statistical techniques

1.37 min per model, respectively. It is noted, structuring data through sequences for the learning process of these models are slightly less than one-step forecast problems in computational cost.

In all ANN structures considered in one-step and multi-step scenarios, pre-processing the MGP series through differentiation resulted in better prediction errors. Similarly, using another regressive variable such as precipitations for the bivariate analysis in both scenarios positively impacted the learning and generalization of the models.

This research helps to describe and predict hydropower generation, particularly in Ecuador. Results obtained with the proposed ANN model can help organize and plan the electric sector, which is important for the energy policy-maker sector. The methodology can also be extended to use in other fields.

Promising results were seen with the seq2seq LSTM architecture but slightly below MLPs. In further studies, we plan to use datasets with a greater number of observations and other recent time series techniques (e.g., latent recurrent neural networks and transformers). In this regard, we will also study the impact of applying super-resolution techniques to increase the granularity of the data set in the performance of recent very deep ANN architectures.

Acknowledgements The authors kindly acknowledge the support from University of Guayaquil. Computational and physical resources were provided by ESPOL. Juan Gómez-Romero is partially supported by the University of Granada and the Spanish Ministries of Science, Innovation and Universities (TIN2017-91223- EXP) and Economy and Competitiveness (TIN2015-64776-C3-1-R).

Funding Funding for open access charge: Universidad de Granada / CBUA. This work has been funded by the Universidad de Guayaquil through the Grant Number FCI-015-2019. This work has also been supported by ESPOL, Grant Number FIMCP-CERA-05-2017.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. IHA (2020) Hydropower Status Report 2020. London
2. Killingtveit Å (2018) Hydropower. In: Letcher T (ed) Managing global warming: an interface of technology and human issues, 1st edn. Academic Press, Durban, pp 265–315
3. Ministerio de Electricidad y Energia Renovable (2016) Plan Maestro de Electricidad 2016–2025, pp 1–440
4. ARCONEL (2019) Estadisticas Anuales Y Multianual Del Sector Eléctrico Ecuatoriano 2018. Quito
5. Zhou F, Li L, Zhang K et al (2020) Forecasting the evolution of hydropower generation. Proc ACM SIGKDD Int Conf Knowl Discov Data Min. https://doi.org/10.1145/3394486.3403337
6. Chen JF, Lo SK, Do QH (2017) Forecasting monthly electricity demands: an application of neural networks trained by heuristic algorithms. Information. https://doi.org/10.3390/info8010031
7. Mite-León M, Barzola-Monteses J (2018) Statistical model for the forecast of hydropower production in Ecuador. Int J Renew Energy Res 10:1130–1137
8. Barzola-Monteses J, Mite-León M, Espinoza-Andaluz M et al (2019) Time series analysis for predicting hydroelectric power production: the Ecuador case. Sustainability 11:1–19. https://doi.org/10.3390/su11236539
9. Abiodun OI, Jantan A, Omolara AE et al (2018) State-of-the-art in artificial neural network applications: a survey. Heliyon 4:e00938. https://doi.org/10.1016/j.heliyon.2018.e00938
10. Abiodun OI, Jantan A, Omolara AE et al (2019) Comprehensive review of artificial neural network applications to pattern recognition. IEEE Access 7:158820–158846. https://doi.org/10.1109/ACCESS.2019.2945545
11. Kostić S, Stojković M, Prohaska S (2016) Hydrological flow rate estimation using artificial neural networks: model development and potential applications. Appl Math Comput 291:373–385. https://doi.org/10.1016/j.amc.2016.07.014
12. Lopes MNG, Da Rocha BRP, Vieira AC et al (2019) Artificial neural networks approaches for predicting the potential for hydropower generation: a case study for Amazon region. J Intell Fuzzy Syst 36:5757–5772. https://doi.org/10.3233/JIFS-181604
13. Torres JF, Hadjout D, Sebaa A et al (2021) Deep learning for time series forecasting: a survey. Big Data 9:3–21. https://doi.org/10.1089/big.2020.0159
14. Amasyali K, El-Gohary NM (2018) A review of data-driven building energy consumption prediction studies. Renew Sustain Energy Rev 81:1192–1205. https://doi.org/10.1016/j.rser.2017.04.095
15. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. The MIT Press, Cambridge
16. Zheng J, Xu C, Zhang Z, Li X (2017) Electric load forecasting in smart grids using long-short-term-memory based recurrent neural network. In: 2017 51st Annual conference on information sciences and systems, CISS 2017, pp 1–6. https://doi.org/10.1109/CISS.2017.7926112
17. Wang X, Zhao T, Liu H, He R (2019) Power consumption predicting and anomaly detection based on long short-term memory neural network. In: 2019 IEEE 4th international conference on cloud computing and big data analytics (ICCCBDA). IEEE, Chengdu, China, pp 487–491
18. Hochreiter S (1997) Long short-term. Memory 1780:1735–1780
19. Barzola-Monteses J, Espinoza-andaluz M, Mite-León M, Flores-Morán M (2020) Energy consumption of a building by using long short-term memory network: a forecasting study. In: 39th International conference of the Chilean computer science society, SCCC 2020. Coquimbo, pp 1–6
20. Peng L, Liu S, Liu R, Wang L (2018) Effective long short-term memory with differential evolution algorithm for electricity price

prediction. Energy 162:1301–1314. https://doi.org/10.1016/j.energy.2018.05.052
21. Sherstinsky A (2020) Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Phys D Nonlinear Phenom 404:132306. https://doi.org/10.1016/j.physd.2019.132306
22. Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: Advances in neural information processing systems 27 (NIPS 2014). Montreal, Canada, pp 3104–3112
23. Sehovac L, Nesen C, Grolinger K (2019) Forecasting building energy consumption with deep learning: a sequence to sequence approach. In: Proceedings of 2019 IEEE international congress on internet of things, ICIOT 2019—Part 2019 IEEE world congress on services, pp 108–116. https://doi.org/10.1109/ICIOT.2019.00029
24. Brownlee J (2018) Machine learning mastery. In: Multi-step LSTM Time Series Forecasting Model. Power usage. https://machinelearningmastery.com/how-to-develop-lstm-models-for-multi-step-time-series-forecasting-of-household-power-consumption/. Accessed 19 Feb 2021
25. Skomski E, Lee JY, Kim W et al (2020) Sequence-to-sequence neural networks for short-term electrical load forecasting in commercial office buildings. Energy Build 226:110350. https://doi.org/10.1016/j.enbuild.2020.110350
26. Jung J, Han H, Kim K, Kim HS (2021) Machine learning-based small hydropower potential prediction under climate change. Energies 14:3643. https://doi.org/10.3390/en14123643
27. Razi M, Yusuff MA, Tee BT, Zakaria KA (2016) Prediction of available power being generate in small hydropower system at Sungai Perting Bentong Pahang. MATEC Web Conf. https://doi.org/10.1051/matecconf/20179001028
28. Oyerinde GT, Wisser D, Hountondji FCC et al (2016) Quantifying uncertainties in modeling climate change impacts on hydropower production. Climate. https://doi.org/10.3390/cli4030034
29. Dehghani M, Riahi-Madvar H, Hooshyaripor F et al (2019) Prediction of hydropower generation using grey Wolf optimization adaptive neuro-fuzzy inference system. Energies. https://doi.org/10.3390/en12020289
30. Tamm O, Luhamaa A, Tamm T (2016) Modeling future changesinthe north-estonian hydropower production by using SWAT. Hydrol Res 47:835–846. https://doi.org/10.2166/nh.2015.018
31. Chen J, Zhong PA (2019) A multi-time-scale power prediction model of hydropower station considering multiple uncertainties. Sci Total Environ 677:612–625. https://doi.org/10.1016/j.scitotenv.2019.04.430
32. Contreras E, Herrero J, Crochemore L et al (2020) Seasonal climate forecast skill assessment for the management of water resources in a run of river hydropower system in the Poqueira River (Southern Spain). Water (Switzerland). https://doi.org/10.3390/W12082119
33. Hidalgo IG, Paredes-Arquiola J, Andreu J et al (2020) Hydropower generation in future climate scenarios. Energy Sustain Dev 59:180–188. https://doi.org/10.1016/j.esd.2020.10.007
34. Farfán JF, Palacios K, Ulloa J, Avilés A (2020) A hybrid neural network-based technique to improve the flow forecasting of physical and data-driven models: methodology and case studies in Andean watersheds. J Hydrol Reg Stud 27:100652. https://doi.org/10.1016/j.ejrh.2019.100652
35. Agencia de Regulación y Control de Electricidad (2019) ARCONEL. Quito—Ecuador
36. INAMHI (2018) Instituto Nacional de Meteorología e Hidrología. http://www.serviciometeorologico.gob.ec/. Accessed 3 Jan 2019
37. Cryer JD, Chan K-S (2008) Time series analysis with applications in R, 2nd edn. Springer, Iowa City
38. Jayalakshmi T, Santhakumaran A (2011) Statistical normalization and back propagation for classification. Int J Comput Theory Eng 3:89–93. https://doi.org/10.7763/ijcte.2011.v3.288
39. Torres J (2018) Deep learning, 2nd edn. Watch this Space, Barcelona
40. Li L, Jamieson K, DeSalvo G et al (2018) Hyperband: a novel bandit-based approach to hyperparameter optimization. J Mach Learn Res 18:1–52
41. Snoek J, Larochelle H, Adams RP (2012) Practical Bayesian optimization of machine learning algorithms. In: NIPS'12: Proceedings of the 25th international conference on neural information processing systems. Lake Tahoe Nevada, pp 2951–2959
42. Bergmeir C, Benítez JM (2012) On the use of cross-validation for time series predictor evaluation. Inf Sci (NY) 191:192–213. https://doi.org/10.1016/j.ins.2011.12.028
43. Nath Lopes M, Lamberts R (2018) Development of a metamodel to predict cooling energy consumption of HVAC systems in office buildings in different climates. Sustainability. https://doi.org/10.3390/su10124718
44. Somu N, MR GR, Ramamritham K (2020) A hybrid model for building energy consumption forecasting using long short term memory networks. Appl Energy 261:1–20. https://doi.org/10.1016/j.apenergy.2019.114131
45. Koprinska I, Wu D, Wang Z (2018) Convolutional neural networks for energy time series forecasting. In: 2018 International joint conference on neural networks (IJCNN). IEEE, pp 1–8