

## Article

# Accelerated Energy Forecasting Models with Metaheuristics for Swift Solutions in Building Management

D. Criado-Ramón <sup>1</sup>, Maria Ruxandra Cojocaru <sup>1</sup>, L. G. B. Ruiz <sup>2</sup>, Lorenzo Servadei <sup>3</sup>, Robert Wille <sup>3</sup>,  
M. P. Cuéllar <sup>1</sup> and M. C. Pegalajar <sup>1,\*</sup>

<sup>1</sup> Department of Computer Science and Artificial Intelligence, University of Granada, C/Periodista Daniel Saucedo Aranda s.n, 18014 Granada, Spain; dcriado@ugr.es (D.C.-R.); ruxandracojocaru@ugr.es (M.R.C.); manupc@ugr.es (M.P.C.)

<sup>2</sup> Department of Software Engineering, University of Granada, C/Periodista Daniel Saucedo Aranda s.n, 18014 Granada, Spain; bacaruiz@ugr.es

<sup>3</sup> Chair for Design Automation, Technical University of Munich, Arcisstraße 21, 80333 Munich, Germany; lorenzo.servadei@tum.de (L.S.); robert.wille@tum.de (R.W.)

\* Correspondence: mcarmen@decsai.ugr.es

## Abstract

This work presents a CUDA-accelerated methodology for training multiple neural networks in parallel using population-based metaheuristics. The goal is to obtain fast and accurate short-term energy-forecasting models for time-sensitive building-management applications. We evaluate five metaheuristic optimizers and their memetic variants, for which a local-search stage based on the *ADAM* optimizer is incorporated. The models are assessed on eleven real-world energy-consumption time series using training time, root mean squared error (RMSE), mean absolute error (MAE), and normalized RMSE (NRMSE). The results show that the proposed memetic approaches are competitive under strict training-time budgets, although unconstrained *ADAM* remains a strong overall reference.

**Keywords:** time series forecasting; metaheuristics; GPU; neural networks

## 1. Introduction

Electricity has become one of the most relevant resources of our time. It is present in a wide variety of electronic devices that we use daily, such as phones, computers, heating/cooling systems, or illumination systems, among many others. As it is such a vital resource for human life, discovering ways to optimally produce and distribute this form of energy in a sustainable and cost-efficient manner has become a global objective supported by many governments and institutions [1].

In order to support this process, short-term load forecasting models are frequently used to make predictions within short periods of time, allowing planners and building managers to make more informed decisions. In building-management applications, these forecasts are not an end in themselves, but rather an input for subsequent operational decisions. For example, in demand-response scenarios, anticipating a consumption peak can give operators enough time to adjust controllable loads, modify schedules, or apply demand-reduction actions. Similarly, in energy purchasing or trading contexts, updated short-term forecasts can help estimate near-future demand and reduce the uncertainty associated with market or procurement decisions.

However, these models still face many challenges, as technological and behavioral changes can substantially modify the expected energy consumption. This can be seen in technologies



Academic Editor: Jiaming Li

Received: 2 June 2026

Revised: 19 July 2026

Accepted: 20 July 2026

Published: 23 July 2026

**Copyright:** © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

such as self-consumption [2], flexible demand response [3], electric vehicles [4], or energy communities [5]. Furthermore, sudden changes in consumer behavior, such as confinement during the COVID-19 crisis [6], can also substantially alter the expected consumption profile. Thus, these models should be able to be trained quickly when no previous forecasting model is available, and re-trained after enough data has been collected to learn new consumption patterns, hence reducing forecasting errors.

Over the last few decades, artificial intelligence (AI) researchers have studied a wide variety of algorithms for energy forecasting tasks. Although earlier models mainly used statistical autoregressive models such as ARIMA [7,8], most recent works in the field make use of some kind of neural network in their design [9], with Long Short-Term Memory (LSTM) neural networks being one of the most popular architectures in recent years. Wang et al. [10] studied the use of different machine learning models to forecast a periodic time series of energy consumption. For this purpose, they used the energy consumption of a refrigerator. The results showed the superiority of Artificial Neural Networks (ANN) over other models, with LSTM being the most prominent model.

Nevertheless, although deep learning models based on these architectures are currently the most popular, recent studies also show that different neural architectures can be useful depending on the specific characteristics of the energy management system [11]. Indeed, several studies have shown that simpler architectures can provide better results in some energy-forecasting scenarios [12–14]. This can be influenced by many factors, such as the forecast horizon, the amount of training data available, and the granularity of the data. A distinct feature of energy consumption data is that it usually only uses a few variables, in many cases only previous energy consumption, and beyond that, the other usual variables that are frequently used are those related to the date and the temperature. Although this represents a lower computational cost than those required for other fields (i.e., computer vision), there is usually not enough data to use all of the GPU cores simultaneously, leading to an inefficient use of the GPU due to the underuse of its computational resources. In this paper, we aim to evaluate the potential of alternative methods for training ANNs, specifically metaheuristic algorithms, that can use GPU computational capabilities to achieve optimized solutions quickly, while avoiding methods that are susceptible to becoming stuck in local minima.

The use of metaheuristic algorithms in ANN training has been a topic of great interest in AI research over the past few decades [15,16], as these algorithms can be used to avoid falling in local optima or to find the optimal hyperparameters for a model [9]. Energy forecasting papers that use metaheuristics to find optimal hyperparameters mainly seek to find an optimal neural network configuration and some parameters that control the training process (optimizer, learning rate). Several representative studies in this area are summarized in Table 1. Bacanin et al. [17] evaluated the use of Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) to select the input size, number of hidden neurons and the optimizer-related hyperparameters in recurrent neural networks. Other works have also explored GA and PSO for related forecasting settings, such as smart-grid demand prediction [18], heating and cooling load prediction [19], and LSTM-based electric-load forecasting [20]. Luo et al. [21] used a GA to optimize the number of layers, hidden neurons, activation function, and optimizer in a deep feedforward neural network. Luo and Oyedele [22] used a GA to optimize the number of layers, hidden units, dropout, and learning rate in an LSTM neural network. Other works have used metaheuristic algorithms to avoid falling in local optima due to their population-based approaches and their balance of exploration and exploitation instead of only exploitation as seen in gradient-based approaches. In [23], a comparison of several metaheuristics was provided to train feedforward ANNs. The results showed that the Teaching–Learning–Based Optimizer provided the best

results, although only a single time series was used. Other comparisons of a similar style have also been carried out in different works. Sahraei and Çodur [24] compared the use of GA, PSO and Simulated Annealing to train a feedforward neural network. Other interesting approaches go beyond just using metaheuristics and propose hardware-accelerated implementations to train these models much faster. Ruiz et al. [25] proposed a parallel CPU implementation of a memetic GA algorithm to train Elman's recurrent neural networks (ENNs). Iruela et al. [26] proposed a parallel GPU implementation of the NSGA-II algorithm to train a multilayer perceptron, allowing for a much faster training time. More recently, Iruela et al. [27] proposed a GPU-accelerated adaptation of PSO for multi-objective optimization of ANN models applied to energy-consumption prediction. However, even though this topic has been previously studied, there is a large gap to be filled, as most works mainly focus on the implementation of one specific metaheuristic algorithm for each specific neural network.

**Table 1.** Representative studies using metaheuristics and neural models for energy-related forecasting.

| Study                    | Application                                | Metaheuristics         | Neural Architecture(s) |
|--------------------------|--------------------------------------------|------------------------|------------------------|
| Muralitharan et al. [18] | Energy-demand prediction                   | GA, PSO                | MLP                    |
| Zhou et al. [19]         | Heating/cooling load prediction            | ABC, PSO               | MLP                    |
| Bouktif et al. [20]      | Electric-load forecasting                  | GA, PSO                | LSTM                   |
| Bacanin et al. [17]      | Multivariate energy generation forecasting | GA, PSO                | LSTM, GRU              |
| Luo et al. [21]          | Hourly electricity consumption             | GA                     | Deep MLP               |
| Luo et al. [22]          | Hyperparameter selection for LSTM          | GA                     | LSTM                   |
| Phatai et al. [23]       | Comparative ANN training                   | Several metaheuristics | MLP                    |
| Sahraei and Çodur [24]   | Energy-demand prediction                   | GA, SA, PSO            | MLP                    |
| Ruiz et al. [25]         | Parallel training of Elman neural network  | Memetic GA             | ENN                    |
| Iruela et al. [26]       | GPU evolutionary training                  | NSGA-II                | MLP                    |
| Iruela et al. [27]       | GPU multi-objective optimization           | PSO                    | MLP                    |

As a consequence, this work aims to address this gap by studying whether several GPU-accelerated metaheuristic and memetic training strategies can provide competitive neural forecasting models within short training times. Unlike previous studies, which have typically focused on a single optimizer, a single neural architecture, or a limited number of time series, this work provides a broader comparison across optimization strategies, neural architectures, and real-world energy-consumption datasets. The main contributions of this work can be summarized as follows:

- We provide an open-source CUDA implementation based on Tensor Cores to accelerate the training of neural energy-forecasting models using five population-based metaheuristics and their corresponding memetic variants, which incorporate a local search stage based on ADAM.

- We evaluate the proposed training strategies across three neural network architectures with different modeling properties: multilayer perceptrons, Elman recurrent neural networks, and long short-term memory networks.
- We assess the proposed approach on 11 time series extracted from two real-world energy-consumption datasets.
- We analyze forecasting performance using multiple error metrics, including RMSE, MAE, and NRMSE, together with training time, in order to characterize the trade-off between predictive performance and computational cost.

The remainder of the paper is structured as follows. Section 2 presents the theoretical background for CUDA, the ANN architectures and the metaheuristic algorithms studied. Section 3 presents the dataset used and explains the proposed approach. Section 4 defines the experiments and provides the parameters required to reproduce them. Section 5 presents and discusses the results obtained for all the time series that were studied. Lastly, Section 6 draws the final conclusions from our research.

## 2. Preliminary Concepts

In this section, we present the theoretical background of ANNs, metaheuristic algorithms and the GPU/CUDA model. While these may be basic concepts for some readers, we consider them important for those who may not be familiar with the topics.

### 2.1. Artificial Neural Networks

ANNs are machine learning models that feature multiple linked computational nodes (neurons), usually structured in layers. During the training process, the weights and biases of the neural network are optimized. In this study, three different types of neural network have been evaluated.

An MLP [28] is a simple and widely used ANN type consisting of an input layer, one or more hidden layers, and an output layer. Each neuron applies a weighted sum with all neurons from the previous layer followed by an activation function  $f$  (Equation (1)), introducing non-linearity into the model.

$$h = f(Wx + b) \quad (1)$$

where  $h$  is a vector that contains all the hidden outputs,  $W$  is the matrix of weights,  $x$  is the input data and  $b$  is the vector of biases.

An ENN [29] is a type of RNN that includes a hidden layer with feedback connections, allowing it to capture temporal dependencies in sequential data. This feedback connection provides a copy of the output of the hidden neuron, which is used as an additional input for the hidden neuron in the next element of the sequence. Equation (2) mathematically describes the computation of the hidden output at time step  $t$  ( $h_t$ ).

$$h_t = g(Wx_t + Uh_{t-1} + b) \quad (2)$$

where  $U$  are the weights for the recurrent connections and  $h_{t-1}$  is the hidden output from the previous time step.

An LSTM [30] is another type of RNN designed to overcome the vanishing gradient problem [31] present in other types of RNN, like the ENN. LSTM hidden neurons feature a hidden state  $h_t$ , a cell state  $c_t$  and three gates used at each time step: the input gate  $i_t$ , which controls incoming information; the forget gate  $f_t$ , which regulates the retention of past information; and the output gate  $o_t$ , which determines the output based on the current input and past hidden states. Each gate and the cell state has its own set of weights  $W$ , recurrent weights  $U$ , and biases  $b$ . This leads to a much more sophisticated architecture

that requires more data and time to be trained but can provide excellent results. The LSTM unit can be described mathematically as follows ( $\otimes$  denotes the element-wise product).

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (3)$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (4)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (5)$$

$$c_t = f_t \cdot c_{t-1} + i_t \otimes g(W_c x_t + U_c h_{t-1} + b_c) \quad (6)$$

$$h_t = o_t \otimes g(c_t) \quad (7)$$

## 2.2. Metaheuristic Algorithms

Metaheuristic algorithms are general optimization approaches usually inspired by natural processes that guide the search for the optimal solution without any problem-specific knowledge. Thus, these algorithms try to either minimize or maximize an objective function, called “fitness”, which for the purposes of this study will be the neural network loss. Five different metaheuristic algorithms were used to optimize the weights and biases of the three neural network architectures studied. PSO was selected as a well-established metaheuristic. Also, we included four promising recently developed algorithms. Given the diverse nature of energy time series and the need for rapid response to unusual events, we believe a comparative analysis of multiple algorithms is essential to achieve optimal performance.

PSO [32] is a heuristic optimization algorithm inspired by the social behavior of animals. In a PSO, a population of potential solutions, represented as particles, is initialized in a solution space. Each particle has a position and velocity, which are adjusted iteratively (Equations (8) and (9)) to explore and exploit the solution space. The movement of each particle is influenced by its own historical best position (personal best) and the best position found by any particle in the entire swarm (global best). This allows particles to balance exploration (searching for new areas in the solution space) and exploitation (refining around known good areas).

$$V_i(t+1) = w \cdot V_i(t) + c_1 \cdot r_1 \cdot (P_i - X_i(t)) + c_2 \cdot r_2 \cdot (G - X_i(t)) \quad (8)$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (9)$$

where

- $X_i(t)$  is the  $i$ -th particle of the swarm at iteration  $t$ .
- $V_i(t)$  is the velocity of the  $i$ -th particle of the swarm at iteration  $t$ .
- $w$  is the inertia weight that controls the impact of the previous velocity.
- $c_1$  and  $c_2$  are acceleration coefficients that control the influence of the personal and global best positions, respectively.
- $r_1$  and  $r_2$  are random values between 0 and 1.
- $P_i$  is the personal best position of particle  $i$ .
- $G$  is the global best position found by any particle in the swarm.

The Equilibrium Optimizer (EO) [33] is a metaheuristic optimization algorithm inspired by the dynamics of mass balance in a control volume. Each individual, called a concentration or particle, is updated in each iteration according to the equation for the conservation of mass in a control volume (Equation (10)). Additionally, a concentration pool is used to preserve the best solutions found during the execution of the algorithm and

contains the four best individuals found and an additional individual that is the average of the four other individuals in the concentration pool.

$$\vec{C} = \vec{C}_{eq} + (\vec{C} - \vec{C}_{eq}) \cdot \vec{F} + \frac{\vec{G}}{\vec{\lambda} V} (1 - \vec{F}) \quad (10)$$

where

- $\vec{C}$  is the current individual.
- $\vec{C}_{eq}$  is a random individual from the concentration pool.
- $\vec{\lambda}$  is a random vector in  $[0, 1]$ .
- $\vec{F}$  is an exponential term based on  $\vec{\lambda}$  that increases with the number of iterations up to 1.
- $\vec{G}$  is the multiplication of  $\vec{F}$  and vector with a random point for each feature between  $\vec{C}_{eq}$  and  $\vec{C}$ .
- $V$  is a constant that represents the volume (usually set to 1).

The Marine Predators Algorithm (MPA) [34] is a metaheuristic that is inspired by the widespread foraging strategies (Lévy and Brownian movements) of ocean predators and their prey. This algorithm updates the strategy of each individual based on the number of iterations that have passed. During the first third of the iterations, the individuals are updated using Brownian movement between the current individual and the best individual. During the second third, half of the population is updated using a Lévy movement and the second half is updated using another Brownian movement weighted by a convergence factor that decreases over time. The Lévy movement is used with the convergence factor for the last third of iterations. Additionally, to avoid local optima, after each iteration, 80% of the population move in their immediate vicinity by adding a random value in their range and the remaining 20% do a longer jump.

The Whale Optimization Algorithm (WOA) [35] is an optimization technique inspired by the cooperative hunting behavior of humpback whales that simulates the process through which whales work together to encircle and capture prey. The algorithm employs two main phases: the encircling prey phase and the searching for prey phase. In the encircling prey phase, solutions (representing whales) adjust their positions to surround the best solution found so far with a 50% chance of using a shrinking mechanism (Equation (11)) and a 50% chance of using a spiral model (Equation (12)). In the searching for prey phase, the solutions explore the search space based on the position of the best global solution. These phases are driven by mathematical equations that model the hunting behavior of whales.

$$\vec{X}(t+1) = \vec{X}^*(t) - \vec{A} \cdot \vec{D} \quad (11)$$

$$\vec{X}(t+1) = \vec{D}' \cdot e^{bl} \cdot \cos(2 \cdot \pi \cdot l) + \vec{X}^*(t) \quad (12)$$

$$\vec{X}(t+1) = \vec{X}_{rand}(t) - \vec{A} \cdot \vec{D}_{rand} \quad (13)$$

where

- $\vec{X}(t)$  is the individual at iteration  $t$
- $\vec{X}^*(t)$  is the best individual found up to iteration  $t$ .
- $\vec{A}$  is a random vector with a linearly decreasing range over the course of the iterations.
- $\vec{D}'$  is the absolute distance between the best individual and the current individual.

- $\vec{D}$  is the multiplication of  $\vec{D'}$  and a random vector in  $[0, 1]$ .
- $b$  is a constant that defines the shape of the spiral and  $l$  is a random number in  $[-1, 1]$ .
- $\vec{X}_{rand}$  is a random whale from the population.
- $\vec{D}_{rand}$  is the equivalent of  $\vec{D}$  using  $\vec{X}_{rand}$  instead of the best individual.

Political Optimizer (PO) [36] is a global optimization algorithm that is inspired by the multiple phases of politics. Unlike other algorithms, each individual in the population has two roles: a party and a constituency. The individual with the best fitness for each party and each constituency is designated as its leader. During each iteration, the political phases are run sequentially. First, the election campaign phase is used to improve the fitness of each individual prior to the election. This is done using a strategy called recent past-based position updating strategy with the previous individual and the best solution. The process is repeated twice, initially with the party leader and subsequently with the constituency winner. Then, the party switching phase occurs, in which there is a random possibility that each member is exchanged with the least fit member of another party. This probability linearly decreases to zero over the course of the iterations. Then, the fitness values of the individual are adapted and, lastly, the parliamentary affairs step takes place. In this step, for each constituency winner, a new individual is computed by randomly taking another constituency winner and adding the absolute distance between the two weighted by a random vector in  $[-1, 1]$ . If the new individual improves the fitness value, it replaces the previous one.

### 2.3. The GPU/CUDA Model

GPUs (Graphics Processing Units) were initially designed to accelerate graphics computation in computers. Their architecture features a large number of cores that operate at a slower latency than CPU cores, making them exceptionally good for massive parallel tasks. Due to this, they have become a key component in advancements in AI, as they can significantly reduce the time required to train AI models.

CUDA (Compute Unified Device Architecture) is the acronym used to name the architecture and general-purpose programming model of NVIDIA's graphics cards. A CUDA-capable GPU features a hierarchical structure of memory and computational components. At its lowest level, the CUDA computational model follows a Single Instruction Multiple Threads (SIMT) model, where 32 contiguous CUDA cores (a "warp") must execute the same instruction simultaneously. Multiple warps are contained together in the multiple "streaming multiprocessors" available inside the GPU. Each streaming multiprocessor can execute code independently, allowing concurrent execution of different GPU applications, since they contain their own CUDA cores, registers, and cache memory, as can be seen in Figure 1, which shows the structure of one of the 142 multiprocessors available in the GPU used and the CUDA memory hierarchy. Additionally, newer GPU generations have additional cores inside each multiprocessor for specific tasks, such as Tensor Cores, designed to accelerate matrix multiplications, which are extremely relevant in many AI algorithms. From a memory perspective, the GPU features a connection to the motherboard memory through the PCIe interface, a large but relatively slow "global memory", and two levels of fast cache: the L2 cache, which is shared across all streaming multiprocessors, and the L1 cache, which is smaller, faster, and local to each streaming multiprocessor.

The CUDA programming model can be used either through some of its specific-purpose libraries (i.e., cuBLAS for linear algebra operations) or by directly writing GPU code through the C/C++ CUDA extension. In this extension of C/C++, the GPU code is written in special functions called "kernels" that contain the set of instructions to be executed by each thread of the GPU. When the kernel is called, the programmer must indicate the total number of threads to use by providing a number of blocks and a number

of threads per block. Each block will be executed in one of the streaming multiprocessors, allowing threads within the same block to cooperate faster as they can directly use the L1 cache and synchronization barriers local to their streaming multiprocessors.

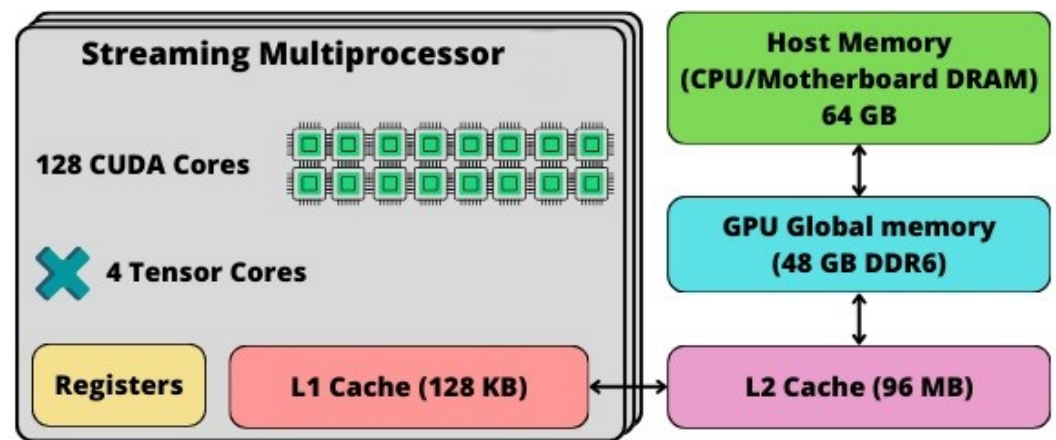


Figure 1. A streaming multiprocessor and the CUDA memory hierarchy.

### 3. Methodology

#### 3.1. Dataset Description

The data for this study were derived from two datasets: the UNICON data set [37] and the energy consumption of Uruguay.

The UNICON dataset is an open-source dataset that provides information on electricity, gas and energy consumption from 71 buildings across 5 different campuses at La Trobe University. It also contains weather information for each campus. For the purposes of our experiment, electricity consumption was the main variable studied, and the temperature was used as an exogenous variable in the forecast. The observations are provided with a granularity of 15 min, with the first observations dating from 2018 to 2021. The dataset containing energy consumption from Uruguay contained only one time series that provided data on energy consumption from 2007 to 2014 and provided information on hourly consumption, temperature and local holidays.

The experiment began with a preprocessing pipeline that started by choosing a subset of the substantial number of buildings accessible in the UNICON dataset. To capture a wide range of building behaviors, we selected the 10 buildings with the largest amount of available data. These buildings, identified by IDs 1, 2, 3, 9, 10, 12, 14, 41, 57 and 60, have significant variations in their time series and therefore include differences in energy consumption patterns, building profiles and environmental factors. In both datasets, all missing values in each building were filled by linear interpolation. Then, each building was treated as an independent experiment and the data from each building was partitioned, with the first 70% used for training and validation and the remaining 30% for testing. The first partition was divided, with the first 70% used for training and the remaining 30% for validation. Lastly, data from each building was scaled from the original range to the [0, 1] range using min–max normalization. The split was always chronological and no shuffling was applied. The min–max normalization parameters were fitted only on the training partition and then applied unchanged to the validation and test partitions, so that neither the validation nor the test observations influenced the scaling parameters.

The selected UNICON buildings cover two campuses and several building categories, as summarized, together with the effective time-series sizes, in Table 2. The forecasting problem was formulated as a direct multi-output prediction of the next 24 h. For UNICON,

this corresponds to the next 96 values, because the sampling interval is 15 min. For Uruguay, it corresponds to the next 24 hourly values.

**Table 2.** Summary of the time series used in the experiments after preprocessing.

| Time Series | Campus         | Category      | Observations | Inputs | Forecast Windows | Train/Test Windows |
|-------------|----------------|---------------|--------------|--------|------------------|--------------------|
| Building 1  | Bundoora       | Other         | 72,192       | 2      | 751              | 525/226            |
| Building 2  | Bundoora       | Other         | 72,192       | 2      | 751              | 525/226            |
| Building 3  | Bundoora       | Other         | 72,192       | 2      | 751              | 525/226            |
| Building 9  | Albury-Wodonga | Teaching      | 116,544      | 2      | 1213             | 849/364            |
| Building 10 | Albury-Wodonga | Teaching      | 116,544      | 2      | 1213             | 849/364            |
| Building 12 | Albury-Wodonga | Other         | 116,544      | 2      | 1213             | 849/364            |
| Building 14 | Albury-Wodonga | Mixed use     | 116,544      | 2      | 1213             | 849/364            |
| Building 41 | Bundoora       | Teaching      | 69,888       | 2      | 727              | 508/219            |
| Building 57 | Bundoora       | Teaching      | 42,912       | 2      | 446              | 312/134            |
| Building 60 | Bundoora       | Mixed use     | 79,008       | 2      | 822              | 575/247            |
| Uruguay     | –              | National load | 70,128       | 2      | 2915             | 2040/875           |

### 3.2. Proposed Approach

#### 3.2.1. CUDA Implementations and Memory Layout

The CUDA implementations presented in this paper can be mainly classified into three distinct categories: Tensor Cores, map-style kernels, and reduction kernels.

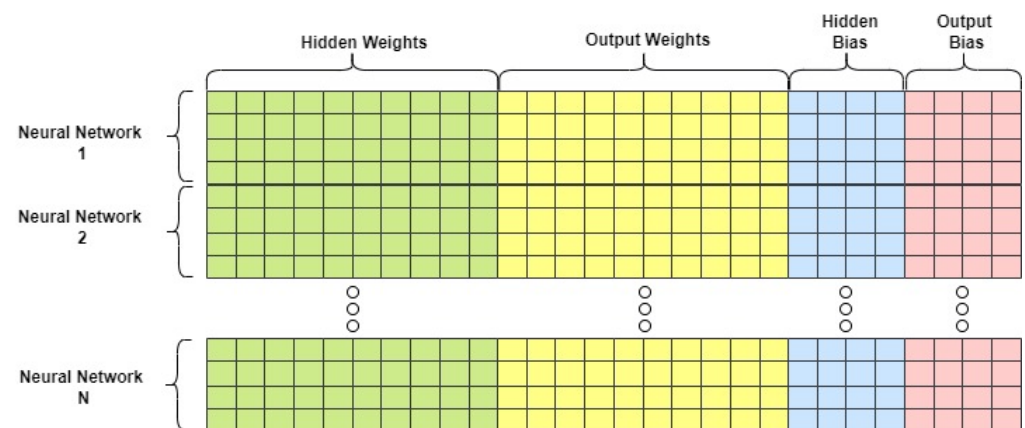
Tensor Cores are used through the cuBLAS library to compute the matrix multiplication operations required in the forward and backward passes of the ANNs as efficiently as possible. An important factor to take into account is that in order to get the most benefit from Tensor Cores there are certain requirements for memory alignment, as otherwise loading the submatrix into the coprocessor may require multiple reads, drastically reducing performance. As shown below, all data structures had their dimensions padded to the next multiple of 8, adding only the zeros required to ensure that the padded data did not affect the results.

Map-style kernels are simple kernels in which a function is applied to all elements in a data structure (i.e., activation functions). In this type of kernel, the computations for each element of the data structure are performed by one thread, allowing the GPU to use as many resources as possible as long as the data structure is large enough. The most important factor in this type of kernel is making sure that adjacent threads access adjacent positions in memory (“coalesced access”), as when a thread requires a memory position, its warp will require a small chunk of memory. Thus, having coalesced access will reduce the total number of memory operations requested by the warp, providing the fastest results.

Lastly, reduction kernels are kernels in which all threads cooperate to compute an aggregation of results, usually a sum or an average of elements in a data structure. For this type of kernel, primarily warp-based reductions were used. In this approach, each warp should perform an independent reduction, and the kernel should start with each thread of the warp performing its own aggregation in a register using coalesced access to the data structure. Afterward, warp-intrinsic instructions can be used to perform reductions in the registers directly without the need for memory access.

The memory layout depicted in Figure 2 was used for the trainable parameters regardless of the ANN and metaheuristic, meaning all metaheuristic algorithms shared the same representation for their individuals. In ANNs that have recurrent weights, the recurrent weights are located between the hidden and the output weights. In this configuration, the dimensions of each substructure, such as hidden weights and output weights, were

adjusted to the nearest multiple of 8 if they were not already a multiple. This strategic padding was designed to optimize the utilization of Tensor Cores, since their primary bottleneck lies in slow data transfer from GPU memory to the Tensor Core. This padding, as specified in Section 2.1.11 of the cuBLAS documentation [38], ensures that the submatrices loaded into the Tensor Cores are read and written as quickly as possible, as they match the dimensions and pointer alignments required for optimal performance. The padded positions were filled with zeros and were excluded from the effective sample count, so they did not modify the value of the objective function.



**Figure 2.** A visual scheme of the memory layout for a MLP architecture.

Moreover, this memory layout is not just optimized for Tensor Core usage; it also aligns with the optimal memory structure for most metaheuristic operators, as most of them will use map-style operations with one or two individuals. Thus, having the features of the individuals adjacent in the memory while each thread computes one of those features will ensure that adjacent threads will access adjacent positions in memory (coalesced memory access), reducing the number of GPU memory accesses to the bare minimum.

### 3.2.2. ANN Models' Adaptation

This section briefly describes the procedure carried out for each of the neural network architectures studied. In all cases, the first step is to prepare the dataset by adding the required padding and loading it into the GPU memory. For the RNN, the dataset is loaded into the memory, making sure that the data of all samples and features for each time step is adjacent in the GPU memory, ensuring coalesced access while processing each time step. Afterwards, depending on the architecture used, different approaches are taken.

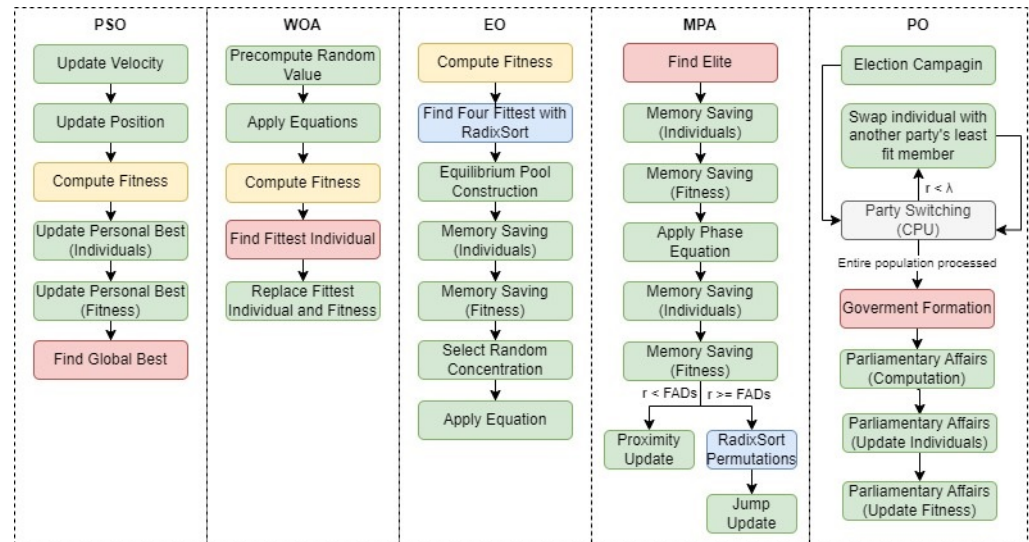
In the case of the MLP, once the dataset has been prepared, Tensor Cores are used to compute the hidden weights with a map-style kernel used to compute the hidden output. Lastly, Tensor Cores are used again to compute the final output and a reduction is applied to compute the fitness of all neural networks.

In the case of the RNNs, both architectures feature a similar approach, as we allowed the GPU to asynchronously compute the operations for the time step inputs and the recurrent inputs. Furthermore, as is common for efficient computation on GPUs [39], all LSTM weights and biases from the different gates were concatenated into a single matrix, making both schemes almost identical, with the exception of the LSTM not only applying the activation function but rather using all element-wise operations in its map-style kernel.

### 3.3. Metaheuristic Implementation Details

Most metaheuristic algorithms require mostly map-style kernels, since most of their operators update the features of an individual according to an element-wise equation.

Figure 3 presents a visual depiction of the implementation details explained in the following section, where  $r$  represents a random number from a uniform distribution in  $[0, 1]$ , green represents a map-style kernel, red represents reduction kernels, yellow represents the use of the forward pass presented in Section 3.2.2, and blue represents the use of other CUDA libraries.



**Figure 3.** A flowchart of the kernels and methods used to implement each metaheuristic in CUDA.

PSO requires the computation of two equations per generation and the use of a memory saving mechanism to preserve the best solutions found by each individual. A generation starts with the computation of the required random values based on a map-style kernel. Then, Equations (8) and (9) are computed, with a map-style kernel used for each. Subsequently, the individual best particle is updated with another map-style kernel, in which the individual best particle is updated if its fitness is better than the previous fitness value. Afterwards, another map-style kernel is launched to update the data structure that holds the individual best fitness, and, lastly, a reduction on the individual best fitness is launched to find the index of the most fit individual (the global best).

WOA has one of the simplest implementations, as only one of the three equations (Equations (11)–(13)) is used on an individual based on random values and some additional computations. Thus, a generation starts with a map-style kernel that computes all random values. Afterwards, another map-style kernel is launched, which applies one of the equations depending on the previously computed random values. Lastly, a reduction is used to find the fittest individual of that generation, which replaces the previous one if it is fitter.

The main feature of EO is the use of the candidate pool, a data structure that preserves the four fittest individuals and another individual that is the average of the previous four. In this algorithm, instead of launching four reductions to find the fittest individual, the fastest GPU sorting algorithm (RadixSort) is used to find them. Once found, a map-style kernel is launched first to construct the candidate pool with the four fittest unique individuals found so far and an additional individual computed as the average of the previous four. Afterwards, a memory saving mechanism similar to the one used in PSO is applied. However, in this case, instead of an additional structure, if the new individual does not improve in fitness, the individual from the previous generation is restored. Then a random array is created to assign to each individual one of the concentrations (an integer from 0 to 4) and a map-style kernel is used to update all individuals in the population, as they all use the same equation (Equation (10)).

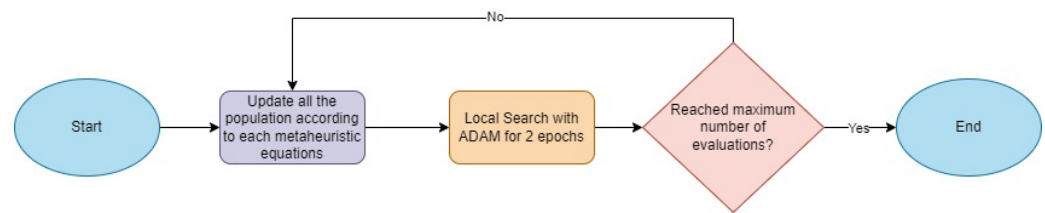
MPA features a more complex strategy, as it requires multiple equations to be computed and uses a memory-saving mechanism twice. A generation starts by finding the fittest individual, called “Elite” in this algorithm, and using the same memory mechanism used in EO. Then, a slightly different equation is applied to the population depending on the current phase. There are three phases, one for each third of the generations used. During the first phase, a Brownian movement update is performed on all individuals. During the second phase, the first half of the population is updated with a Brownian movement with a convergence factor, and the remaining with a Lévy movement. In the last phase, all individuals are updated with a Lévy movement with a convergence factor. Each phase is implemented with its own map-style kernel. Subsequently, the memory-saving procedure is applied again. After this, one of two equations is applied to all individuals in the generation based on a random number in  $[0, 1]$ . If the random number is below the *FADs* parameter (0.2), a map-style kernel is used to explore positions in the proximity of each individual. In the other case, another map-style kernel is used to cause a larger movement. This movement is computed by taking the distance between two randomly selected individuals. Thus, before the computation of that kernel, we must compute two random permutations of arrays containing indexes for the individuals. These are computed by applying RadixSort on a randomly generated array and using the sorted indexes as the desired permutations.

PO is the last and most complex implementation, as it has several phases and one of its components has data dependencies that force sequential execution. The election campaign, which updates the position of each individual according to a set of equations, is done with a simple map-style kernel. Afterward, the party switching starts, in which each individual may be randomly swapped with the least fit individual of another randomly selected party, which creates a data dependency. Thus, this part is mainly done by the CPU with the GPU only using a map-style kernel to exchange those individuals and their fitness values. The next step is the government formation, which designates the most fit individuals per party and constituency as leaders. This is done with a reduction-style kernel, as it implies the finding of the indexes with the best fitness value. The last step is the parliamentary affairs. In this step, each constituency leader is attracted to a randomly selected individual, which is implemented in a map-style kernel. Lastly, if this improves the fitness value, the individuals and their fitness values are updated with two more map-style kernels.

### 3.4. Memetic Algorithms

In addition to the metaheuristic algorithms, a memetic algorithm was also implemented for each of the proposed metaheuristics. Memetic algorithms are optimization approaches that blend metaheuristic algorithms with local search strategies applied to each individual in the population after the metaheuristic operators. Thus, memetic algorithms provide a combination of global exploration and local exploitation that aims to enhance the efficiency of finding optimal solutions. The general scheme for the memetic algorithm used can be seen in Figure 4, where a local search is performed using backpropagation with the *ADAM* optimizer and applied to each individual in the population. However, in PO, since it has a much higher population size, only eight individuals perform the local search with *ADAM*, as otherwise it does not have enough evaluations to converge.

The backpropagation process is implemented according to each neural network architecture, using Tensor Cores for every operation that could be expressed as a matrix multiplication; map-style kernels to apply derivatives of activation functions and update the weights and biases; and reduction-style kernels to aggregate the gradients from all the samples processed.



**Figure 4.** A summary of the memetic algorithm used for all metaheuristics.

#### 4. Experiments

The forecasting task in which all algorithms were evaluated was to predict the next 24 h starting at 0:00 (next 96/24 values, depending on the data set). After a preliminary trial-and-error study in TensorFlow, the optimal number of previous time steps used as input for the forecast (“lags”) was set to 96 (1 day of data) for the UNICON dataset and 168 (7 days of data) for the Uruguay dataset, and the optimal activation functions were set to sigmoid for the MLP and the default hyperbolic tangent for the ENN and LSTM neural networks. In all architectures, temperature was also taken as an exogenous variable and was used as an additional input to the neural network. The number of hidden neurons was treated as a shared architectural parameter across all three neural models. For both the proposed approach and the TensorFlow implementations, the MLP, ENN, and LSTM configurations were systematically evaluated from 1 to 90 hidden neurons.

In metaheuristic and memetic algorithms, all weights and biases were randomly initialized using a uniform distribution in  $[-1, 1]$ . As is common in the literature, to make the comparison fair between metaheuristic and memetic algorithms, the algorithms were limited to a maximum number of evaluations of the objective function (30,000).

Additionally, a baseline with the *ADAM* optimizer using the GPU in TensorFlow was also compared. Since it uses a completely different approach than the metaheuristic algorithms, two different versions were evaluated. *ADAM<sub>T</sub>* will refer to a time-limited execution of *ADAM* that will stop if, after processing a batch, more time is spent than the slowest memetic or metaheuristic algorithm for that complexity (same number of hidden neurons and neural network architecture). This will be our main focus of comparison, as it allows us to compare the use of *ADAM* and the other algorithms in the time-restricted cases that are of our interest, such as energy trading, demand-response or better grid management through quicker adaptation to sudden changes.

In addition to our main experiment, we also implemented an unlimited version of *ADAM*, referred to as *ADAM<sub>U</sub>*, which will continue running until it achieves the optimal results. Thus, after setting the parameters via trial and error, *ADAM<sub>U</sub>* was set to run for up to 100 epochs with a batch size of 32 with Early Stopping if the validation loss did not improve over 5 epochs. This provided enough epochs to guarantee convergence and helped mitigate any overfitting issues. The weights and biases in the *ADAM* baselines were initialized with the default methods used in TensorFlow.

All experiments were executed on an NVIDIA RTX 6000 Ada Generation GPU using 32-bit floating-point precision (FP32). The implementation was compiled with CUDA 12.5 and leveraged the cuBLAS and cuRAND libraries. The reported training times encompass the entire optimization process including bidirectional host-device data transfers.

The CUDA source code required to reproduce the experiments is available at [40].

All experimentation was repeated a total of 5 times, using seeds 1996 to 2000, and most of the metaheuristic parameters were initialized to the values suggested by their authors, as can be seen in Table 3.

**Table 3.** List of parameters for each algorithm.

| Algorithm | Parameter       | Values                     |
|-----------|-----------------|----------------------------|
| EO        | Population size | 30                         |
|           | $a_1$           | 2                          |
|           | $a_2$           | 1                          |
|           | $g^p$           | 0.5                        |
| PSO       | Population size | 30                         |
|           | $c_1$           | 1.5                        |
|           | $c_2$           | 2.5                        |
|           | $W$             | 0.6                        |
| PO        | $n$             | 8                          |
|           | $\lambda$       | 1 to 0 linearly decreasing |
| WOA       | Population size | 30                         |
| MPA       | Population size | 25                         |
|           | $FADs$          | 0.2                        |
|           | $P$             | 0.5                        |
| ADAM      | learning rate   | 0.001                      |

Alongside the time each algorithm took to train the neural networks, we used the Root Mean Squared Error (RMSE) as the main quality metric. The RMSE is an error metric frequently used in time-series forecasting and regression problems that gives a lower error with more accurate predictions and penalizes larger errors. Mathematically, the RMSE is computed as indicated in Equation (14).

$$RMSE = \sqrt{\frac{1}{N} \cdot \sum_{i=1}^N (\hat{y}_i - y_i)^2} \quad (14)$$

where  $\hat{y}_i$  is the forecasted value,  $y_i$  is the expected value and  $N$  is the sample size. RMSE is complemented with the Mean Absolute Error (MAE), which is less sensitive to large individual errors, and the Normalized Root Mean Squared Error (NRMSE), which facilitates comparisons between time series with different consumption scales. These metrics are defined in Equations (15) and (16).

$$MAE = \frac{1}{N} \cdot \sum_{i=1}^N |\hat{y}_i - y_i| \quad (15)$$

$$NRMSE = \frac{RMSE}{y_{\max} - y_{\min}} \quad (16)$$

In Equation (16),  $y_{\max}$  and  $y_{\min}$  are the maximum and minimum values of the corresponding test target series. RMSE, MAE and NRMSE are reported as mean  $\pm$  sample standard deviation across the five independent seeds.

## 5. Results

This section presents the results obtained from the experiments. Initially, we evaluate the performance of the proposed models. Then, the computational efficiency of the algorithms is assessed, with a particular focus on the time required to converge to optimal solutions. It should be noted that, in order to perform a fair comparison between the metaheuristic algorithms and the  $ADAM_T$  algorithm, the computational time is fixed. This allows for a direct evaluation of the algorithms' performance, independent of their nature. The architecture-wise tables also include the unconstrained  $ADAM_U$  reference, which makes it possible to distinguish between the matched-time comparison and the behavior of a conventional early-stopped ADAM training process. All tables report the mean and sample standard deviation over the five independent seeds. In addition to RMSE, MAE

and NRMSE are included, where NRMSE is computed by dividing RMSE by the range of the corresponding test target.

Table 4 outlines the results for the most studied neural network architecture, the MLP. This is also one of the most interesting architectures for time-restricted scenarios, as it is usually extremely fast to train. Tables 4–6 report the best hidden neuron setting for each algorithm and time series, selected according to the lowest mean RMSE. The same selected configuration is then reported with MAE and NRMSE. In these architecture-wise tables,  $ADAM_T$  is retained as the ADAM reference because it follows the same time-budgeted comparison criterion as the metaheuristic runs.

**Table 4.** Results for the MLP architecture.

| Time Series | Algorithm | Hidden | RMSE                   | MAE                    | NRMSE                  |
|-------------|-----------|--------|------------------------|------------------------|------------------------|
| Building 1  | $ADAM_T$  | 1      | 0.3845 ± 0.0050        | 0.2964 ± 0.0077        | 0.4794 ± 0.0062        |
|             | $ADAM_U$  | 75     | <b>0.1148 ± 0.0004</b> | <b>0.0832 ± 0.0008</b> | <b>0.1432 ± 0.0004</b> |
|             | EO        | 34     | 0.1530 ± 0.0018        | 0.1162 ± 0.0017        | 0.1908 ± 0.0023        |
|             | EO (M)    | 7      | 0.1303 ± 0.0020        | 0.1018 ± 0.0030        | 0.1624 ± 0.0025        |
|             | MPA       | 21     | 0.2970 ± 0.2321        | 0.2275 ± 0.1791        | 0.3703 ± 0.2895        |
|             | MPA (M)   | 42     | 0.3395 ± 0.2846        | 0.2558 ± 0.2150        | 0.4233 ± 0.3549        |
|             | PO        | 2      | 0.1594 ± 0.0031        | 0.1170 ± 0.0039        | 0.1987 ± 0.0038        |
|             | PO (M)    | 86     | 0.1804 ± 0.0440        | 0.1353 ± 0.0379        | 0.2250 ± 0.0549        |
|             | PSO       | 1      | 0.3186 ± 0.0307        | 0.2251 ± 0.0207        | 0.3972 ± 0.0383        |
|             | PSO (M)   | 75     | 0.1469 ± 0.0049        | 0.1087 ± 0.0024        | 0.1832 ± 0.0061        |
|             | WOA       | 7      | 0.2615 ± 0.0040        | 0.2450 ± 0.0070        | 0.3261 ± 0.0050        |
|             | WOA (M)   | 12     | 0.1337 ± 0.0099        | 0.1016 ± 0.0097        | 0.1668 ± 0.0124        |
| Building 2  | $ADAM_T$  | 35     | 0.2556 ± 0.0037        | 0.2092 ± 0.0047        | 0.2803 ± 0.0040        |
|             | $ADAM_U$  | 86     | <b>0.2495 ± 0.0055</b> | <b>0.1994 ± 0.0041</b> | <b>0.2735 ± 0.0060</b> |
|             | EO        | 11     | 0.2731 ± 0.0085        | 0.2300 ± 0.0041        | 0.2995 ± 0.0094        |
|             | EO (M)    | 42     | 0.2594 ± 0.0048        | 0.2216 ± 0.0019        | 0.2844 ± 0.0053        |
|             | MPA       | 16     | 0.4428 ± 0.2542        | 0.3573 ± 0.2029        | 0.4855 ± 0.2788        |
|             | MPA (M)   | 12     | 0.3664 ± 0.1503        | 0.2961 ± 0.1144        | 0.4017 ± 0.1648        |
|             | PO        | 2      | 0.3063 ± 0.0107        | 0.2522 ± 0.0060        | 0.3359 ± 0.0117        |
|             | PO (M)    | 6      | 0.2973 ± 0.0053        | 0.2456 ± 0.0033        | 0.3260 ± 0.0058        |
|             | PSO       | 1      | 0.3359 ± 0.0340        | 0.2660 ± 0.0290        | 0.3683 ± 0.0373        |
|             | PSO (M)   | 77     | 0.3391 ± 0.0489        | 0.2608 ± 0.0347        | 0.3718 ± 0.0536        |
|             | WOA       | 4      | 0.3800 ± 0.0009        | 0.3338 ± 0.0011        | 0.4166 ± 0.0009        |
|             | WOA (M)   | 22     | 0.2635 ± 0.0020        | 0.2158 ± 0.0010        | 0.2889 ± 0.0022        |
| Building 3  | $ADAM_T$  | 1      | 1.0030 ± 0.0080        | 0.9133 ± 0.0073        | 0.6249 ± 0.0050        |
|             | $ADAM_U$  | 81     | 0.2958 ± 0.0254        | 0.2296 ± 0.0216        | 0.1843 ± 0.0159        |
|             | EO        | 11     | 0.3098 ± 0.0041        | 0.2369 ± 0.0036        | 0.1930 ± 0.0025        |
|             | EO (M)    | 9      | 0.2888 ± 0.0081        | 0.2203 ± 0.0058        | 0.1799 ± 0.0050        |
|             | MPA       | 1      | 0.4878 ± 0.2004        | 0.3640 ± 0.1384        | 0.3039 ± 0.1249        |
|             | MPA (M)   | 1      | 0.4011 ± 0.1283        | 0.2946 ± 0.0818        | 0.2499 ± 0.0799        |
|             | PO        | 1      | 0.3165 ± 0.0025        | 0.2441 ± 0.0028        | 0.1972 ± 0.0016        |
|             | PO (M)    | 80     | 0.3628 ± 0.0676        | 0.2800 ± 0.0538        | 0.2261 ± 0.0421        |
|             | PSO       | 1      | 0.3922 ± 0.0166        | 0.2989 ± 0.0112        | 0.2444 ± 0.0103        |
|             | PSO (M)   | 78     | 0.3202 ± 0.0093        | 0.2430 ± 0.0060        | 0.1995 ± 0.0058        |
|             | WOA       | 1      | 0.4253 ± 0.0016        | 0.3469 ± 0.0014        | 0.2650 ± 0.0010        |
|             | WOA (M)   | 81     | <b>0.2710 ± 0.0029</b> | <b>0.2015 ± 0.0026</b> | <b>0.1688 ± 0.0018</b> |

Table 4. Cont.

| Time Series | Algorithm | Hidden | RMSE                                  | MAE                                   | NRMSE                                 |
|-------------|-----------|--------|---------------------------------------|---------------------------------------|---------------------------------------|
| Building 9  | $ADAM_T$  | 1      | $2.0609 \pm 0.0299$                   | $1.5356 \pm 0.0295$                   | $0.3972 \pm 0.0058$                   |
|             | $ADAM_U$  | 4      | <b><math>1.1237 \pm 0.0082</math></b> | $0.8072 \pm 0.0069$                   | <b><math>0.2166 \pm 0.0016</math></b> |
|             | EO        | 49     | $1.1364 \pm 0.0021$                   | $0.8860 \pm 0.0031$                   | $0.2190 \pm 0.0004$                   |
|             | EO (M)    | 9      | $1.1249 \pm 0.0037$                   | <b><math>0.7721 \pm 0.0081</math></b> | $0.2168 \pm 0.0007$                   |
|             | MPA       | 18     | $1.7562 \pm 1.1950$                   | $1.3856 \pm 0.9179$                   | $0.3384 \pm 0.2303$                   |
|             | MPA (M)   | 11     | $1.8143 \pm 0.9383$                   | $1.3261 \pm 0.7461$                   | $0.3496 \pm 0.1808$                   |
|             | PO        | 1      | $1.2295 \pm 0.0188$                   | $0.9631 \pm 0.0138$                   | $0.2369 \pm 0.0036$                   |
|             | PO (M)    | 5      | $1.2026 \pm 0.0116$                   | $0.9366 \pm 0.0092$                   | $0.2318 \pm 0.0022$                   |
|             | PSO       | 1      | $1.7190 \pm 0.0885$                   | $1.2407 \pm 0.0516$                   | $0.3313 \pm 0.0171$                   |
|             | PSO (M)   | 51     | $1.6881 \pm 0.7040$                   | $1.3053 \pm 0.5753$                   | $0.3253 \pm 0.1357$                   |
|             | WOA       | 11     | $1.3540 \pm 0.0087$                   | $1.1710 \pm 0.0126$                   | $0.2609 \pm 0.0017$                   |
|             | WOA (M)   | 83     | $1.1343 \pm 0.0101$                   | $0.7802 \pm 0.0155$                   | $0.2186 \pm 0.0019$                   |
| Building 10 | $ADAM_T$  | 1      | $0.5281 \pm 0.0183$                   | $0.4174 \pm 0.0128$                   | $0.2450 \pm 0.0085$                   |
|             | $ADAM_U$  | 66     | <b><math>0.2007 \pm 0.0030</math></b> | <b><math>0.1371 \pm 0.0024</math></b> | <b><math>0.0931 \pm 0.0014</math></b> |
|             | EO        | 4      | $0.2483 \pm 0.0056$                   | $0.1845 \pm 0.0047$                   | $0.1152 \pm 0.0026$                   |
|             | EO (M)    | 5      | $0.2131 \pm 0.0022$                   | $0.1497 \pm 0.0018$                   | $0.0988 \pm 0.0010$                   |
|             | MPA       | 11     | $0.4512 \pm 0.3511$                   | $0.3503 \pm 0.2731$                   | $0.2093 \pm 0.1628$                   |
|             | MPA (M)   | 10     | $0.4345 \pm 0.3073$                   | $0.3205 \pm 0.2346$                   | $0.2015 \pm 0.1425$                   |
|             | PO        | 1      | $0.2620 \pm 0.0092$                   | $0.1932 \pm 0.0075$                   | $0.1215 \pm 0.0043$                   |
|             | PO (M)    | 10     | $0.2571 \pm 0.0091$                   | $0.1941 \pm 0.0073$                   | $0.1193 \pm 0.0042$                   |
|             | PSO       | 7      | $0.5955 \pm 0.1420$                   | $0.4234 \pm 0.1138$                   | $0.2762 \pm 0.0658$                   |
|             | PSO (M)   | 72     | $0.4371 \pm 0.4183$                   | $0.3319 \pm 0.3257$                   | $0.2027 \pm 0.1940$                   |
|             | WOA       | 69     | $0.2996 \pm 0.0015$                   | $0.2267 \pm 0.0006$                   | $0.1390 \pm 0.0007$                   |
|             | WOA (M)   | 22     | $0.2063 \pm 0.0043$                   | $0.1453 \pm 0.0053$                   | $0.0957 \pm 0.0020$                   |
| Building 12 | $ADAM_T$  | 1      | $1.4723 \pm 0.0266$                   | $1.1192 \pm 0.0234$                   | $0.3606 \pm 0.0065$                   |
|             | $ADAM_U$  | 65     | <b><math>0.7232 \pm 0.0038</math></b> | <b><math>0.5027 \pm 0.0046</math></b> | <b><math>0.1771 \pm 0.0009</math></b> |
|             | EO        | 42     | $0.7838 \pm 0.0034$                   | $0.6259 \pm 0.0039$                   | $0.1920 \pm 0.0008$                   |
|             | EO (M)    | 6      | $0.7338 \pm 0.0067$                   | $0.5268 \pm 0.0124$                   | $0.1797 \pm 0.0016$                   |
|             | MPA       | 10     | $1.2537 \pm 0.8416$                   | $0.9791 \pm 0.6371$                   | $0.3071 \pm 0.2061$                   |
|             | MPA (M)   | 12     | $1.2243 \pm 0.6759$                   | $0.9167 \pm 0.5394$                   | $0.2999 \pm 0.1655$                   |
|             | PO        | 3      | $0.8298 \pm 0.0143$                   | $0.6466 \pm 0.0096$                   | $0.2032 \pm 0.0035$                   |
|             | PO (M)    | 5      | $0.8193 \pm 0.0079$                   | $0.6461 \pm 0.0111$                   | $0.2007 \pm 0.0019$                   |
|             | PSO       | 6      | $1.4657 \pm 0.2926$                   | $1.0467 \pm 0.1805$                   | $0.3590 \pm 0.0717$                   |
|             | PSO (M)   | 86     | $1.2442 \pm 0.8602$                   | $0.9604 \pm 0.6563$                   | $0.3047 \pm 0.2107$                   |
|             | WOA       | 7      | $0.9194 \pm 0.0085$                   | $0.7605 \pm 0.0097$                   | $0.2252 \pm 0.0021$                   |
|             | WOA (M)   | 12     | $0.7414 \pm 0.0206$                   | $0.5284 \pm 0.0236$                   | $0.1816 \pm 0.0050$                   |
| Building 14 | $ADAM_T$  | 1      | $9.4891 \pm 0.1678$                   | $8.7719 \pm 0.1026$                   | $0.4313 \pm 0.0076$                   |
|             | $ADAM_U$  | 90     | <b><math>2.1826 \pm 0.0181</math></b> | <b><math>1.5399 \pm 0.0241</math></b> | <b><math>0.0992 \pm 0.0008</math></b> |
|             | EO        | 27     | $2.7377 \pm 0.0152$                   | $2.1849 \pm 0.0156$                   | $0.1244 \pm 0.0007$                   |
|             | EO (M)    | 5      | $2.2620 \pm 0.0258$                   | $1.6280 \pm 0.0384$                   | $0.1028 \pm 0.0012$                   |
|             | MPA       | 12     | $6.2957 \pm 5.3753$                   | $4.8802 \pm 3.9984$                   | $0.2862 \pm 0.2443$                   |
|             | MPA (M)   | 12     | $6.8327 \pm 6.2216$                   | $5.2704 \pm 4.9052$                   | $0.3106 \pm 0.2828$                   |
|             | PO        | 4      | $3.1430 \pm 0.3461$                   | $2.4087 \pm 0.2201$                   | $0.1429 \pm 0.0157$                   |
|             | PO (M)    | 4      | $2.8039 \pm 0.0874$                   | $2.1823 \pm 0.0955$                   | $0.1275 \pm 0.0040$                   |
|             | PSO       | 1      | $7.5044 \pm 1.2827$                   | $5.1588 \pm 0.5963$                   | $0.3411 \pm 0.0583$                   |
|             | PSO (M)   | 69     | $5.8660 \pm 3.5970$                   | $4.5801 \pm 2.8207$                   | $0.2666 \pm 0.1635$                   |
|             | WOA       | 11     | $3.1507 \pm 0.0191$                   | $2.5301 \pm 0.0104$                   | $0.1432 \pm 0.0009$                   |
|             | WOA (M)   | 12     | $2.3727 \pm 0.1419$                   | $1.7255 \pm 0.1309$                   | $0.1079 \pm 0.0065$                   |

Table 4. Cont.

| Time Series | Algorithm | Hidden | RMSE                    | MAE                     | NRMSE                  |
|-------------|-----------|--------|-------------------------|-------------------------|------------------------|
| Building 41 | $ADAM_T$  | 1      | 17.7124 ± 0.1994        | 13.9317 ± 0.2295        | 0.4341 ± 0.0049        |
|             | $ADAM_U$  | 80     | 6.9385 ± 0.0959         | 5.4881 ± 0.0544         | 0.1700 ± 0.0023        |
|             | EO        | 8      | 8.9717 ± 0.1461         | 6.7450 ± 0.0986         | 0.2199 ± 0.0036        |
|             | EO (M)    | 31     | 7.0360 ± 0.1230         | 5.5743 ± 0.0906         | 0.1724 ± 0.0030        |
|             | MPA       | 10     | 12.5078 ± 6.0503        | 9.5807 ± 4.7366         | 0.3065 ± 0.1483        |
|             | MPA (M)   | 59     | 18.4419 ± 15.8635       | 14.6962 ± 12.7607       | 0.4519 ± 0.3888        |
|             | PO        | 6      | 9.9704 ± 0.0764         | 7.3421 ± 0.0816         | 0.2443 ± 0.0019        |
|             | PO (M)    | 87     | 10.1977 ± 0.9154        | 7.7724 ± 0.7888         | 0.2499 ± 0.0224        |
|             | PSO       | 1      | 13.5092 ± 1.3011        | 9.7489 ± 0.6298         | 0.3311 ± 0.0319        |
|             | PSO (M)   | 90     | 9.2222 ± 0.2124         | 6.9873 ± 0.1844         | 0.2260 ± 0.0052        |
|             | WOA       | 62     | 11.3373 ± 0.0953        | 9.0111 ± 0.0987         | 0.2778 ± 0.0023        |
|             | WOA (M)   | 54     | <b>6.7485 ± 0.2332</b>  | <b>5.2510 ± 0.2229</b>  | <b>0.1654 ± 0.0057</b> |
| Building 57 | $ADAM_T$  | 1      | 13.9855 ± 0.2353        | 10.1258 ± 0.2988        | 0.3529 ± 0.0059        |
|             | $ADAM_U$  | 79     | <b>7.4302 ± 0.0432</b>  | <b>5.5718 ± 0.0518</b>  | <b>0.1875 ± 0.0011</b> |
|             | EO        | 74     | 8.2659 ± 0.0469         | 6.4898 ± 0.0505         | 0.2086 ± 0.0012        |
|             | EO (M)    | 5      | 7.8127 ± 0.1598         | 5.8604 ± 0.1225         | 0.1972 ± 0.0040        |
|             | MPA       | 1      | 10.1887 ± 1.7993        | 7.7401 ± 1.1227         | 0.2571 ± 0.0454        |
|             | MPA (M)   | 13     | 12.8839 ± 7.0937        | 9.7932 ± 5.3728         | 0.3251 ± 0.1790        |
|             | PO        | 2      | 8.3753 ± 0.1627         | 6.4866 ± 0.1569         | 0.2114 ± 0.0041        |
|             | PO (M)    | 2      | 8.3293 ± 0.0632         | 6.5821 ± 0.0935         | 0.2102 ± 0.0016        |
|             | PSO       | 18     | 18.7006 ± 4.4761        | 13.3461 ± 3.4173        | 0.4719 ± 0.1130        |
|             | PSO (M)   | 85     | 8.3645 ± 0.2402         | 6.5518 ± 0.1677         | 0.2111 ± 0.0061        |
|             | WOA       | 50     | 9.5907 ± 0.2230         | 7.9411 ± 0.1643         | 0.2420 ± 0.0056        |
|             | WOA (M)   | 83     | 7.7682 ± 0.2200         | 5.8580 ± 0.2308         | 0.1960 ± 0.0056        |
| Building 60 | $ADAM_T$  | 1      | 7.9166 ± 0.2864         | 6.4428 ± 0.1865         | 0.2262 ± 0.0082        |
|             | $ADAM_U$  | 4      | 3.2324 ± 0.0233         | 2.0935 ± 0.0215         | 0.0924 ± 0.0007        |
|             | EO        | 14     | 3.9145 ± 0.0276         | 2.5394 ± 0.0403         | 0.1118 ± 0.0008        |
|             | EO (M)    | 2      | <b>3.1694 ± 0.0138</b>  | <b>2.0706 ± 0.0203</b>  | <b>0.0906 ± 0.0004</b> |
|             | MPA       | 3      | 6.4898 ± 4.4778         | 4.6076 ± 3.3408         | 0.1854 ± 0.1279        |
|             | MPA (M)   | 7      | 6.9870 ± 5.1454         | 4.9227 ± 3.9112         | 0.1996 ± 0.1470        |
|             | PO        | 2      | 4.1157 ± 0.1164         | 2.5919 ± 0.0976         | 0.1176 ± 0.0033        |
|             | PO (M)    | 3      | 4.0637 ± 0.2052         | 2.6130 ± 0.1549         | 0.1161 ± 0.0059        |
|             | PSO       | 8      | 8.0054 ± 2.3640         | 5.9222 ± 1.6575         | 0.2287 ± 0.0675        |
|             | PSO (M)   | 66     | 4.1180 ± 0.0920         | 2.8412 ± 0.0501         | 0.1177 ± 0.0026        |
|             | WOA       | 8      | 4.2709 ± 0.0091         | 2.5428 ± 0.0134         | 0.1220 ± 0.0003        |
|             | WOA (M)   | 4      | 3.2601 ± 0.0298         | 2.1210 ± 0.0490         | 0.0931 ± 0.0009        |
| Uruguay     | $ADAM_T$  | 59     | 175.3795 ± 47.1836      | 139.7339 ± 40.5609      | 0.1415 ± 0.0381        |
|             | $ADAM_U$  | 44     | <b>78.0887 ± 7.0743</b> | <b>57.0153 ± 6.2761</b> | <b>0.0630 ± 0.0057</b> |
|             | EO        | 18     | 126.6056 ± 5.1539       | 96.9375 ± 4.1772        | 0.1021 ± 0.0042        |
|             | EO (M)    | 11     | 115.4051 ± 5.6076       | 88.2634 ± 4.8364        | 0.0931 ± 0.0045        |
|             | MPA       | 3      | 216.4235 ± 136.0608     | 165.5939 ± 98.6799      | 0.1746 ± 0.1097        |
|             | MPA (M)   | 2      | 168.2956 ± 77.3559      | 134.3443 ± 68.1895      | 0.1357 ± 0.0624        |
|             | PO        | 4      | 145.0604 ± 14.2275      | 113.3741 ± 11.8625      | 0.1170 ± 0.0115        |
|             | PO (M)    | 1      | 132.5613 ± 4.7039       | 99.9411 ± 2.8500        | 0.1069 ± 0.0038        |
|             | PSO       | 25     | 185.7694 ± 9.3357       | 143.9343 ± 7.1620       | 0.1498 ± 0.0075        |
|             | PSO (M)   | 51     | 140.4087 ± 6.2133       | 108.3580 ± 5.7288       | 0.1133 ± 0.0050        |
|             | WOA       | 2      | 231.0332 ± 2.4518       | 186.7446 ± 2.6248       | 0.1863 ± 0.0020        |
|             | WOA (M)   | 57     | 108.0997 ± 4.5734       | 81.6276 ± 3.8691        | 0.0872 ± 0.0037        |

Note: Bold values indicate the best results for each metric and time series.

In the MLP case,  $ADAM_U$  obtains the best result in eight of the eleven time series, although its optimization requires significantly more time than the metaheuristic approaches, as will be seen later. In particular,  $ADAM_U$  starts to obtain better results as the complexity of the neural network grows. Nevertheless, the results obtained by the proposed metaheuristic and memetic training methods are not dominated uniformly: WOA (M) provides the lowest RMSE in Buildings 3 and 41, and EO (M) does so in Building 60. If  $ADAM_U$  is excluded, the comparison becomes clearly favorable to the memetic algorithms: EO (M) and WOA (M) jointly obtain the best result in ten of the eleven series, while  $ADAM_T$  remains the best option only in Building 2. When the comparison is restricted to the metaheuristic and memetic families, the best method is always memetic. Therefore, for MLP models, the memetic variant is consistently beneficial, although the unconstrained ADAM baseline remains the strongest overall reference if the time to select the architecture and train is not limited.

Table 5 presents the RMSE results for the ENN neural network architecture. The ENN results show a different pattern.  $ADAM_T$  is the best method in seven of the eleven time series, whereas  $ADAM_U$  obtains the best result in two cases and EO obtains the best result in Buildings 41 and 57. Excluding  $ADAM_U$ , and in contrast with the MLP case, the memetic variants do not show a systematic advantage over their base metaheuristics: the best non-ADAM method is a standard metaheuristic in most series, and EO is by far the most consistent non-ADAM alternative. This discrepancy is primarily attributed to certain individuals in the population encountering a common issue seen in RNNs, known as the vanishing/exploding gradient problem. In this situation, large gradients can destabilize backpropagation through many time steps, leading to excessively large updates of the weights and biases.

**Table 5.** Results for the ENN architecture.

| Time Series | Algorithm | Hidden | RMSE                   | MAE                    | NRMSE                  |
|-------------|-----------|--------|------------------------|------------------------|------------------------|
| Building 1  | $ADAM_T$  | 85     | <b>0.1222 ± 0.0038</b> | <b>0.0907 ± 0.0033</b> | <b>0.1523 ± 0.0047</b> |
|             | $ADAM_U$  | 8      | 0.1542 ± 0.0032        | 0.1130 ± 0.0035        | 0.1922 ± 0.0040        |
|             | EO        | 53     | 0.1539 ± 0.0046        | 0.1104 ± 0.0057        | 0.1919 ± 0.0057        |
|             | EO (M)    | 28     | 0.4492 ± 0.1041        | 0.3448 ± 0.0808        | 0.5601 ± 0.1299        |
|             | MPA       | 12     | 0.2902 ± 0.2239        | 0.2232 ± 0.1779        | 0.3618 ± 0.2792        |
|             | MPA (M)   | 1      | 0.2117 ± 0.0556        | 0.1530 ± 0.0399        | 0.2640 ± 0.0694        |
|             | PO        | 2      | 0.1575 ± 0.0021        | 0.1162 ± 0.0031        | 0.1964 ± 0.0026        |
|             | PO (M)    | 2      | 0.1598 ± 0.0061        | 0.1141 ± 0.0052        | 0.1992 ± 0.0076        |
|             | PSO       | 1      | 0.2898 ± 0.0396        | 0.2023 ± 0.0311        | 0.3613 ± 0.0494        |
|             | PSO (M)   | 1      | 0.3426 ± 0.0513        | 0.2646 ± 0.0466        | 0.4271 ± 0.0640        |
|             | WOA       | 5      | 0.2614 ± 0.0073        | 0.2401 ± 0.0121        | 0.3260 ± 0.0091        |
|             | WOA (M)   | 2      | 0.2749 ± 0.0909        | 0.2167 ± 0.0780        | 0.3427 ± 0.1133        |
| Building 2  | $ADAM_T$  | 78     | <b>0.2627 ± 0.0045</b> | <b>0.2168 ± 0.0044</b> | <b>0.2881 ± 0.0049</b> |
|             | $ADAM_U$  | 79     | 0.2700 ± 0.0138        | 0.2175 ± 0.0109        | 0.2961 ± 0.0151        |
|             | EO        | 18     | 0.2885 ± 0.0047        | 0.2306 ± 0.0054        | 0.3163 ± 0.0051        |
|             | EO (M)    | 24     | 0.6124 ± 0.0253        | 0.4921 ± 0.0287        | 0.6715 ± 0.0277        |
|             | MPA       | 13     | 0.4317 ± 0.2235        | 0.3431 ± 0.1653        | 0.4733 ± 0.2450        |
|             | MPA (M)   | 7      | 0.5112 ± 0.2024        | 0.4307 ± 0.1507        | 0.5606 ± 0.2219        |
|             | PO        | 2      | 0.2962 ± 0.0050        | 0.2398 ± 0.0074        | 0.3248 ± 0.0055        |
|             | PO (M)    | 3      | 0.2974 ± 0.0072        | 0.2409 ± 0.0078        | 0.3261 ± 0.0079        |
|             | PSO       | 2      | 0.3631 ± 0.0184        | 0.2961 ± 0.0133        | 0.3981 ± 0.0202        |
|             | PSO (M)   | 1      | 0.4432 ± 0.0226        | 0.3527 ± 0.0205        | 0.4860 ± 0.0248        |
|             | WOA       | 3      | 0.3800 ± 0.0040        | 0.3303 ± 0.0045        | 0.4167 ± 0.0044        |
|             | WOA (M)   | 2      | 0.3109 ± 0.0430        | 0.2636 ± 0.0379        | 0.3410 ± 0.0472        |

Table 5. Cont.

| Time Series | Algorithm | Hidden | RMSE                   | MAE                    | NRMSE                  |
|-------------|-----------|--------|------------------------|------------------------|------------------------|
| Building 3  | $ADAM_T$  | 88     | <b>0.2957 ± 0.0085</b> | <b>0.2261 ± 0.0073</b> | <b>0.1842 ± 0.0053</b> |
|             | $ADAM_U$  | 89     | 0.3115 ± 0.0071        | 0.2411 ± 0.0070        | 0.1941 ± 0.0044        |
|             | EO        | 3      | 0.3079 ± 0.0043        | 0.2327 ± 0.0071        | 0.1919 ± 0.0027        |
|             | EO (M)    | 26     | 1.4496 ± 0.4880        | 1.1650 ± 0.4375        | 0.9032 ± 0.3040        |
|             | MPA       | 1      | 0.4143 ± 0.0817        | 0.3065 ± 0.0460        | 0.2581 ± 0.0509        |
|             | MPA (M)   | 1      | 0.5280 ± 0.2646        | 0.4188 ± 0.2313        | 0.3290 ± 0.1649        |
|             | PO        | 3      | 0.3202 ± 0.0052        | 0.2413 ± 0.0065        | 0.1995 ± 0.0033        |
|             | PO (M)    | 3      | 0.3105 ± 0.0060        | 0.2312 ± 0.0084        | 0.1935 ± 0.0038        |
|             | PSO       | 2      | 0.4819 ± 0.0793        | 0.3624 ± 0.0513        | 0.3003 ± 0.0494        |
|             | PSO (M)   | 1      | 0.6750 ± 0.1655        | 0.5551 ± 0.1582        | 0.4205 ± 0.1031        |
|             | WOA       | 4      | 0.4542 ± 0.0494        | 0.3695 ± 0.0361        | 0.2830 ± 0.0308        |
|             | WOA (M)   | 1      | 0.3589 ± 0.0872        | 0.2874 ± 0.0913        | 0.2236 ± 0.0543        |
| Building 9  | $ADAM_T$  | 58     | <b>1.1517 ± 0.0095</b> | 0.8375 ± 0.0120        | <b>0.2219 ± 0.0018</b> |
|             | $ADAM_U$  | 29     | 1.1678 ± 0.0264        | <b>0.8309 ± 0.0250</b> | 0.2251 ± 0.0051        |
|             | EO        | 25     | 1.2497 ± 0.0461        | 0.9533 ± 0.0358        | 0.2408 ± 0.0089        |
|             | EO (M)    | 25     | 2.9360 ± 0.2329        | 2.3403 ± 0.2271        | 0.5658 ± 0.0449        |
|             | MPA       | 14     | 1.9955 ± 1.5287        | 1.5972 ± 1.2526        | 0.3846 ± 0.2946        |
|             | MPA (M)   | 1      | 1.4010 ± 0.2852        | 1.1383 ± 0.1727        | 0.2700 ± 0.0550        |
|             | PO        | 4      | 1.2261 ± 0.0141        | 0.9722 ± 0.0131        | 0.2363 ± 0.0027        |
|             | PO (M)    | 4      | 1.2439 ± 0.0183        | 0.9901 ± 0.0173        | 0.2397 ± 0.0035        |
|             | PSO       | 2      | 1.4794 ± 0.1058        | 1.1674 ± 0.0462        | 0.2851 ± 0.0204        |
|             | PSO (M)   | 1      | 1.8863 ± 0.2012        | 1.4751 ± 0.1482        | 0.3635 ± 0.0388        |
|             | WOA       | 26     | 1.5022 ± 0.0524        | 1.2041 ± 0.0289        | 0.2895 ± 0.0101        |
|             | WOA (M)   | 2      | 1.2091 ± 0.0196        | 0.9669 ± 0.0040        | 0.2330 ± 0.0038        |
| Building 10 | $ADAM_T$  | 69     | <b>0.2146 ± 0.0048</b> | <b>0.1493 ± 0.0023</b> | <b>0.0995 ± 0.0022</b> |
|             | $ADAM_U$  | 20     | 0.2366 ± 0.0150        | 0.1664 ± 0.0123        | 0.1098 ± 0.0069        |
|             | EO        | 75     | 0.2381 ± 0.0050        | 0.1746 ± 0.0033        | 0.1104 ± 0.0023        |
|             | EO (M)    | 25     | 0.8405 ± 0.3328        | 0.6973 ± 0.3145        | 0.3898 ± 0.1544        |
|             | MPA       | 7      | 0.4406 ± 0.3378        | 0.3337 ± 0.2543        | 0.2043 ± 0.1567        |
|             | MPA (M)   | 2      | 0.6638 ± 0.2011        | 0.5557 ± 0.1293        | 0.3079 ± 0.0933        |
|             | PO        | 3      | 0.2674 ± 0.0030        | 0.2028 ± 0.0050        | 0.1240 ± 0.0014        |
|             | PO (M)    | 2      | 0.2682 ± 0.0051        | 0.2033 ± 0.0067        | 0.1244 ± 0.0024        |
|             | PSO       | 3      | 0.5141 ± 0.0895        | 0.3667 ± 0.0395        | 0.2385 ± 0.0415        |
|             | PSO (M)   | 1      | 0.6141 ± 0.0324        | 0.4759 ± 0.0326        | 0.2849 ± 0.0150        |
|             | WOA       | 1      | 0.2960 ± 0.0013        | 0.2247 ± 0.0008        | 0.1373 ± 0.0006        |
|             | WOA (M)   | 1      | 0.4031 ± 0.3183        | 0.3383 ± 0.3108        | 0.1869 ± 0.1476        |
| Building 12 | $ADAM_T$  | 38     | <b>0.7339 ± 0.0067</b> | <b>0.5113 ± 0.0097</b> | <b>0.1797 ± 0.0016</b> |
|             | $ADAM_U$  | 86     | 0.7704 ± 0.0432        | 0.5398 ± 0.0299        | 0.1887 ± 0.0106        |
|             | EO        | 82     | 0.8088 ± 0.0090        | 0.6071 ± 0.0091        | 0.1981 ± 0.0022        |
|             | EO (M)    | 25     | 1.9827 ± 0.7386        | 1.6066 ± 0.6804        | 0.4856 ± 0.1809        |
|             | MPA       | 14     | 1.5208 ± 1.4211        | 1.1925 ± 1.1413        | 0.3725 ± 0.3480        |
|             | MPA (M)   | 1      | 1.0287 ± 0.2800        | 0.8167 ± 0.1999        | 0.2519 ± 0.0686        |
|             | PO        | 3      | 0.8228 ± 0.0091        | 0.6416 ± 0.0117        | 0.2015 ± 0.0022        |
|             | PO (M)    | 2      | 0.8261 ± 0.0107        | 0.6357 ± 0.0178        | 0.2023 ± 0.0026        |
|             | PSO       | 3      | 1.1186 ± 0.0773        | 0.8388 ± 0.0254        | 0.2740 ± 0.0189        |
|             | PSO (M)   | 1      | 1.4541 ± 0.1689        | 1.1415 ± 0.1684        | 0.3561 ± 0.0414        |
|             | WOA       | 1      | 0.9222 ± 0.0039        | 0.7630 ± 0.0042        | 0.2259 ± 0.0010        |
|             | WOA (M)   | 1      | 0.8032 ± 0.0065        | 0.6432 ± 0.0190        | 0.1967 ± 0.0016        |

Table 5. Cont.

| Time Series | Algorithm | Hidden | RMSE                                  | MAE                                   | NRMSE                                 |
|-------------|-----------|--------|---------------------------------------|---------------------------------------|---------------------------------------|
| Building 14 | $ADAM_T$  | 89     | <b>2.4010 <math>\pm</math> 0.0451</b> | <b>1.7471 <math>\pm</math> 0.0324</b> | <b>0.1091 <math>\pm</math> 0.0020</b> |
|             | $ADAM_U$  | 89     | 2.4363 $\pm$ 0.0838                   | 1.7789 $\pm$ 0.0621                   | 0.1107 $\pm$ 0.0038                   |
|             | EO        | 37     | 2.6143 $\pm$ 0.1158                   | 1.9698 $\pm$ 0.0753                   | 0.1188 $\pm$ 0.0053                   |
|             | EO (M)    | 25     | 11.6575 $\pm$ 0.8991                  | 9.7324 $\pm$ 0.6463                   | 0.5299 $\pm$ 0.0409                   |
|             | MPA       | 3      | 5.7096 $\pm$ 4.1489                   | 4.4029 $\pm$ 3.3011                   | 0.2595 $\pm$ 0.1886                   |
|             | MPA (M)   | 1      | 4.6053 $\pm$ 2.3603                   | 3.5904 $\pm$ 1.6894                   | 0.2093 $\pm$ 0.1073                   |
|             | PO        | 2      | 2.6408 $\pm$ 0.0603                   | 1.8986 $\pm$ 0.0667                   | 0.1200 $\pm$ 0.0027                   |
|             | PO (M)    | 1      | 2.6687 $\pm$ 0.1788                   | 2.0432 $\pm$ 0.2174                   | 0.1213 $\pm$ 0.0081                   |
|             | PSO       | 2      | 5.0752 $\pm$ 1.6974                   | 3.6639 $\pm$ 0.8227                   | 0.2307 $\pm$ 0.0772                   |
|             | PSO (M)   | 1      | 8.3207 $\pm$ 2.1100                   | 6.5812 $\pm$ 1.9882                   | 0.3782 $\pm$ 0.0959                   |
|             | WOA       | 2      | 3.1148 $\pm$ 0.0336                   | 2.5138 $\pm$ 0.0299                   | 0.1416 $\pm$ 0.0015                   |
|             | WOA (M)   | 1      | 3.0179 $\pm$ 0.1309                   | 2.3012 $\pm$ 0.1986                   | 0.1372 $\pm$ 0.0060                   |
| Building 41 | $ADAM_T$  | 87     | 8.9787 $\pm$ 0.1733                   | 6.9928 $\pm$ 0.2207                   | 0.2200 $\pm$ 0.0042                   |
|             | $ADAM_U$  | 60     | 9.2650 $\pm$ 0.1957                   | 7.1558 $\pm$ 0.2528                   | 0.2271 $\pm$ 0.0048                   |
|             | EO        | 70     | <b>8.9736 <math>\pm</math> 0.1611</b> | <b>6.6650 <math>\pm</math> 0.1145</b> | <b>0.2199 <math>\pm</math> 0.0039</b> |
|             | EO (M)    | 28     | 18.2035 $\pm$ 1.9731                  | 14.4382 $\pm$ 2.0421                  | 0.4461 $\pm$ 0.0484                   |
|             | MPA       | 28     | 20.0333 $\pm$ 19.7891                 | 15.3690 $\pm$ 15.4044                 | 0.4909 $\pm$ 0.4850                   |
|             | MPA (M)   | 1      | 11.5144 $\pm$ 2.1832                  | 9.0720 $\pm$ 1.3727                   | 0.2822 $\pm$ 0.0535                   |
|             | PO        | 2      | 10.0180 $\pm$ 0.1658                  | 7.4957 $\pm$ 0.0959                   | 0.2455 $\pm$ 0.0041                   |
|             | PO (M)    | 1      | 10.0720 $\pm$ 0.2301                  | 7.4846 $\pm$ 0.2237                   | 0.2468 $\pm$ 0.0056                   |
|             | PSO       | 1      | 12.0014 $\pm$ 0.6062                  | 9.0111 $\pm$ 0.3041                   | 0.2941 $\pm$ 0.0149                   |
|             | PSO (M)   | 1      | 16.1068 $\pm$ 4.3400                  | 12.9178 $\pm$ 3.9901                  | 0.3947 $\pm$ 0.1064                   |
|             | WOA       | 87     | 14.3964 $\pm$ 0.8007                  | 11.2691 $\pm$ 0.6320                  | 0.3528 $\pm$ 0.0196                   |
|             | WOA (M)   | 2      | 9.8002 $\pm$ 0.2718                   | 7.4622 $\pm$ 0.2445                   | 0.2402 $\pm$ 0.0067                   |
| Building 57 | $ADAM_T$  | 28     | 8.7699 $\pm$ 0.3260                   | 7.0543 $\pm$ 0.2109                   | 0.2213 $\pm$ 0.0082                   |
|             | $ADAM_U$  | 32     | 8.5350 $\pm$ 0.3572                   | 6.7543 $\pm$ 0.4463                   | 0.2154 $\pm$ 0.0090                   |
|             | EO        | 7      | <b>8.2454 <math>\pm</math> 0.1275</b> | <b>6.5214 <math>\pm</math> 0.0889</b> | <b>0.2081 <math>\pm</math> 0.0032</b> |
|             | EO (M)    | 33     | 19.2044 $\pm$ 2.6709                  | 15.4027 $\pm$ 2.1282                  | 0.4847 $\pm$ 0.0674                   |
|             | MPA       | 10     | 14.5993 $\pm$ 12.8251                 | 11.1397 $\pm$ 9.4895                  | 0.3684 $\pm$ 0.3237                   |
|             | MPA (M)   | 1      | 10.0475 $\pm$ 2.5088                  | 8.1396 $\pm$ 1.5571                   | 0.2536 $\pm$ 0.0633                   |
|             | PO        | 2      | 8.5720 $\pm$ 0.2191                   | 6.6639 $\pm$ 0.1960                   | 0.2163 $\pm$ 0.0055                   |
|             | PO (M)    | 1      | 8.4338 $\pm$ 0.1000                   | 6.6123 $\pm$ 0.0262                   | 0.2128 $\pm$ 0.0025                   |
|             | PSO       | 1      | 11.0943 $\pm$ 1.6229                  | 8.4312 $\pm$ 0.8675                   | 0.2800 $\pm$ 0.0410                   |
|             | PSO (M)   | 2      | 21.7337 $\pm$ 7.6042                  | 18.0910 $\pm$ 6.7599                  | 0.5485 $\pm$ 0.1919                   |
|             | WOA       | 8      | 9.6620 $\pm$ 0.1251                   | 8.1283 $\pm$ 0.1226                   | 0.2438 $\pm$ 0.0032                   |
|             | WOA (M)   | 1      | 8.6538 $\pm$ 0.7880                   | 6.8867 $\pm$ 0.7649                   | 0.2184 $\pm$ 0.0199                   |
| Building 60 | $ADAM_T$  | 60     | 3.4174 $\pm$ 0.1289                   | 2.2136 $\pm$ 0.1119                   | 0.0976 $\pm$ 0.0037                   |
|             | $ADAM_U$  | 36     | <b>3.2930 <math>\pm</math> 0.1666</b> | <b>2.0420 <math>\pm</math> 0.1584</b> | <b>0.0941 <math>\pm</math> 0.0048</b> |
|             | EO        | 75     | 3.6854 $\pm$ 0.0988                   | 2.4657 $\pm$ 0.0721                   | 0.1053 $\pm$ 0.0028                   |
|             | EO (M)    | 28     | 9.0090 $\pm$ 1.5905                   | 7.1522 $\pm$ 1.7119                   | 0.2574 $\pm$ 0.0454                   |
|             | MPA       | 15     | 10.8604 $\pm$ 14.4906                 | 8.2647 $\pm$ 11.5617                  | 0.3103 $\pm$ 0.4140                   |
|             | MPA (M)   | 1      | 7.8568 $\pm$ 3.0011                   | 6.4447 $\pm$ 2.4258                   | 0.2245 $\pm$ 0.0857                   |
|             | PO        | 2      | 3.8689 $\pm$ 0.1731                   | 2.5639 $\pm$ 0.1335                   | 0.1105 $\pm$ 0.0049                   |
|             | PO (M)    | 1      | 3.9740 $\pm$ 0.3565                   | 2.5949 $\pm$ 0.2355                   | 0.1135 $\pm$ 0.0102                   |
|             | PSO       | 2      | 7.0210 $\pm$ 1.4784                   | 4.9784 $\pm$ 0.6448                   | 0.2006 $\pm$ 0.0422                   |
|             | PSO (M)   | 1      | 10.9424 $\pm$ 1.6556                  | 8.4999 $\pm$ 1.6327                   | 0.3126 $\pm$ 0.0473                   |
|             | WOA       | 32     | 5.1929 $\pm$ 0.4466                   | 3.7097 $\pm$ 0.4496                   | 0.1484 $\pm$ 0.0128                   |
|             | WOA (M)   | 1      | 4.8196 $\pm$ 1.0955                   | 3.5073 $\pm$ 1.2160                   | 0.1377 $\pm$ 0.0313                   |

Table 5. Cont.

| Time Series | Algorithm | Hidden | RMSE                    | MAE                     | NRMSE                  |
|-------------|-----------|--------|-------------------------|-------------------------|------------------------|
| Uruguay     | $ADAM_T$  | 1      | 282.3086 ± 60.6192      | 230.4367 ± 51.5230      | 0.2277 ± 0.0489        |
|             | $ADAM_U$  | 34     | <b>88.3909 ± 7.8369</b> | <b>65.5108 ± 6.5362</b> | <b>0.0713 ± 0.0063</b> |
|             | EO        | 7      | 113.0601 ± 8.2917       | 87.6163 ± 6.4768        | 0.0912 ± 0.0067        |
|             | EO (M)    | 2      | 236.6443 ± 72.8314      | 197.6232 ± 65.8361      | 0.1909 ± 0.0587        |
|             | MPA       | 1      | 200.6955 ± 68.0416      | 150.3126 ± 41.7739      | 0.1619 ± 0.0549        |
|             | MPA (M)   | 1      | 207.7392 ± 55.0281      | 162.4460 ± 46.7405      | 0.1676 ± 0.0444        |
|             | PO        | 1      | 125.4225 ± 30.6979      | 97.9672 ± 25.6779       | 0.1012 ± 0.0248        |
|             | PO (M)    | 3      | 129.0506 ± 12.1901      | 99.0059 ± 9.4712        | 0.1041 ± 0.0098        |
|             | PSO       | 3      | 173.4970 ± 20.6707      | 131.7044 ± 14.7885      | 0.1399 ± 0.0167        |
|             | PSO (M)   | 1      | 375.9120 ± 96.1482      | 323.7635 ± 87.2623      | 0.3032 ± 0.0776        |
|             | WOA       | 3      | 220.0027 ± 6.8665       | 174.2942 ± 10.1217      | 0.1775 ± 0.0055        |
|             | WOA (M)   | 1      | 433.9186 ± 485.5382     | 347.8444 ± 392.4840     | 0.3500 ± 0.3916        |

Note: Bold values indicate the best results for each metric and time series.

Table 6 presents the results for the LSTM architecture. While  $ADAM_T$  remains the best-performing method in most cases (8 out of 11), metaheuristic and memetic algorithms obtain better results in three time series. This is particularly noteworthy considering the higher dimensionality and architectural complexity of LSTMs compared with ENNs. Despite this increased complexity, the memetic variants again improve the results provided by the plain metaheuristic algorithms in several cases. This is likely related to the fact that LSTMs were specifically designed to mitigate the vanishing-gradient problem by using a cell state that enables error signals to propagate more stably through time. Among the metaheuristic and memetic algorithms, the memetic version of WOA consistently provided the best results.

Table 6. Results for the LSTM architecture.

| Time Series | Algorithm | Hidden | RMSE                   | MAE                    | NRMSE                  |
|-------------|-----------|--------|------------------------|------------------------|------------------------|
| Building 1  | $ADAM_T$  | 86     | <b>0.1137 ± 0.0009</b> | <b>0.0782 ± 0.0021</b> | <b>0.1418 ± 0.0011</b> |
|             | $ADAM_U$  | 83     | 0.1427 ± 0.0020        | 0.1097 ± 0.0036        | 0.1779 ± 0.0025        |
|             | EO        | 50     | 0.1520 ± 0.0043        | 0.1110 ± 0.0040        | 0.1896 ± 0.0054        |
|             | EO (M)    | 24     | 0.1457 ± 0.0027        | 0.1092 ± 0.0033        | 0.1817 ± 0.0033        |
|             | MPA       | 9      | 0.2154 ± 0.0740        | 0.1616 ± 0.0627        | 0.2686 ± 0.0922        |
|             | MPA (M)   | 25     | 0.2550 ± 0.1520        | 0.2001 ± 0.1210        | 0.3180 ± 0.1896        |
|             | PO        | 4      | 0.1618 ± 0.0047        | 0.1156 ± 0.0040        | 0.2017 ± 0.0059        |
|             | PO (M)    | 45     | 0.1518 ± 0.0038        | 0.1090 ± 0.0019        | 0.1893 ± 0.0047        |
|             | PSO       | 1      | 0.2407 ± 0.0280        | 0.1631 ± 0.0114        | 0.3001 ± 0.0349        |
|             | PSO (M)   | 22     | 0.1883 ± 0.0326        | 0.1399 ± 0.0298        | 0.2348 ± 0.0406        |
|             | WOA       | 43     | 0.2995 ± 0.0241        | 0.2538 ± 0.0146        | 0.3734 ± 0.0300        |
|             | WOA (M)   | 63     | 0.1304 ± 0.0034        | 0.0964 ± 0.0051        | 0.1626 ± 0.0042        |

Table 6. Cont.

| Time Series | Algorithm | Hidden | RMSE                                  | MAE                                   | NRMSE                                 |
|-------------|-----------|--------|---------------------------------------|---------------------------------------|---------------------------------------|
| Building 2  | $ADAM_T$  | 79     | <b>0.2510 <math>\pm</math> 0.0043</b> | <b>0.1961 <math>\pm</math> 0.0023</b> | <b>0.2752 <math>\pm</math> 0.0047</b> |
|             | $ADAM_U$  | 86     | 0.2698 $\pm$ 0.0041                   | 0.2172 $\pm$ 0.0037                   | 0.2958 $\pm$ 0.0045                   |
|             | EO        | 32     | 0.2838 $\pm$ 0.0051                   | 0.2292 $\pm$ 0.0037                   | 0.3112 $\pm$ 0.0056                   |
|             | EO (M)    | 48     | 0.2859 $\pm$ 0.0084                   | 0.2276 $\pm$ 0.0064                   | 0.3135 $\pm$ 0.0092                   |
|             | MPA       | 19     | 0.4115 $\pm$ 0.1936                   | 0.3328 $\pm$ 0.1507                   | 0.4512 $\pm$ 0.2123                   |
|             | MPA (M)   | 14     | 0.3247 $\pm$ 0.0731                   | 0.2616 $\pm$ 0.0586                   | 0.3560 $\pm$ 0.0802                   |
|             | PO        | 1      | 0.2950 $\pm$ 0.0087                   | 0.2387 $\pm$ 0.0109                   | 0.3234 $\pm$ 0.0095                   |
|             | PO (M)    | 33     | 0.2881 $\pm$ 0.0036                   | 0.2310 $\pm$ 0.0032                   | 0.3159 $\pm$ 0.0040                   |
|             | PSO       | 16     | 0.3447 $\pm$ 0.0289                   | 0.2782 $\pm$ 0.0202                   | 0.3780 $\pm$ 0.0317                   |
|             | PSO (M)   | 28     | 0.3215 $\pm$ 0.0376                   | 0.2588 $\pm$ 0.0253                   | 0.3525 $\pm$ 0.0412                   |
|             | WOA       | 43     | 0.4196 $\pm$ 0.0621                   | 0.3480 $\pm$ 0.0396                   | 0.4601 $\pm$ 0.0681                   |
|             | WOA (M)   | 42     | 0.2579 $\pm$ 0.0101                   | 0.2139 $\pm$ 0.0057                   | 0.2828 $\pm$ 0.0111                   |
| Building 3  | $ADAM_T$  | 73     | <b>0.2631 <math>\pm</math> 0.0048</b> | <b>0.1935 <math>\pm</math> 0.0047</b> | <b>0.1639 <math>\pm</math> 0.0030</b> |
|             | $ADAM_U$  | 84     | 0.3093 $\pm$ 0.0098                   | 0.2407 $\pm$ 0.0083                   | 0.1927 $\pm$ 0.0061                   |
|             | EO        | 3      | 0.2999 $\pm$ 0.0037                   | 0.2238 $\pm$ 0.0051                   | 0.1868 $\pm$ 0.0023                   |
|             | EO (M)    | 3      | 0.2837 $\pm$ 0.0060                   | 0.2055 $\pm$ 0.0032                   | 0.1767 $\pm$ 0.0037                   |
|             | MPA       | 10     | 0.5312 $\pm$ 0.2575                   | 0.4202 $\pm$ 0.2126                   | 0.3309 $\pm$ 0.1605                   |
|             | MPA (M)   | 9      | 0.4503 $\pm$ 0.2267                   | 0.3311 $\pm$ 0.1751                   | 0.2806 $\pm$ 0.1413                   |
|             | PO        | 11     | 0.3196 $\pm$ 0.0052                   | 0.2440 $\pm$ 0.0042                   | 0.1991 $\pm$ 0.0032                   |
|             | PO (M)    | 48     | 0.3089 $\pm$ 0.0079                   | 0.2320 $\pm$ 0.0049                   | 0.1925 $\pm$ 0.0049                   |
|             | PSO       | 20     | 0.4028 $\pm$ 0.0400                   | 0.3127 $\pm$ 0.0311                   | 0.2510 $\pm$ 0.0249                   |
|             | PSO (M)   | 31     | 0.3758 $\pm$ 0.0606                   | 0.2861 $\pm$ 0.0472                   | 0.2341 $\pm$ 0.0377                   |
|             | WOA       | 7      | 0.5112 $\pm$ 0.0303                   | 0.4162 $\pm$ 0.0247                   | 0.3185 $\pm$ 0.0189                   |
|             | WOA (M)   | 14     | 0.2704 $\pm$ 0.0034                   | 0.1956 $\pm$ 0.0053                   | 0.1685 $\pm$ 0.0021                   |
| Building 9  | $ADAM_T$  | 72     | 1.1405 $\pm$ 0.0295                   | 0.8166 $\pm$ 0.0091                   | 0.2198 $\pm$ 0.0057                   |
|             | $ADAM_U$  | 69     | 1.1453 $\pm$ 0.0092                   | 0.8229 $\pm$ 0.0147                   | 0.2207 $\pm$ 0.0018                   |
|             | EO        | 42     | 1.1541 $\pm$ 0.0115                   | 0.8638 $\pm$ 0.0117                   | 0.2224 $\pm$ 0.0022                   |
|             | EO (M)    | 21     | 1.1266 $\pm$ 0.0127                   | 0.8286 $\pm$ 0.0016                   | 0.2171 $\pm$ 0.0024                   |
|             | MPA       | 12     | 1.4342 $\pm$ 0.3881                   | 1.1166 $\pm$ 0.2672                   | 0.2764 $\pm$ 0.0748                   |
|             | MPA (M)   | 15     | 1.5457 $\pm$ 0.5821                   | 1.1420 $\pm$ 0.4547                   | 0.2979 $\pm$ 0.1122                   |
|             | PO        | 6      | 1.2184 $\pm$ 0.0117                   | 0.9623 $\pm$ 0.0065                   | 0.2348 $\pm$ 0.0023                   |
|             | PO (M)    | 1      | 1.2036 $\pm$ 0.0146                   | 0.9461 $\pm$ 0.0232                   | 0.2320 $\pm$ 0.0028                   |
|             | PSO       | 4      | 1.3789 $\pm$ 0.0552                   | 1.1009 $\pm$ 0.0483                   | 0.2657 $\pm$ 0.0106                   |
|             | PSO (M)   | 40     | 1.2732 $\pm$ 0.0727                   | 1.0091 $\pm$ 0.0805                   | 0.2454 $\pm$ 0.0140                   |
|             | WOA       | 5      | 1.3700 $\pm$ 0.0233                   | 1.1599 $\pm$ 0.0225                   | 0.2640 $\pm$ 0.0045                   |
|             | WOA (M)   | 38     | <b>1.1136 <math>\pm</math> 0.0070</b> | <b>0.8134 <math>\pm</math> 0.0158</b> | <b>0.2146 <math>\pm</math> 0.0013</b> |
| Building 10 | $ADAM_T$  | 30     | <b>0.2060 <math>\pm</math> 0.0024</b> | <b>0.1390 <math>\pm</math> 0.0021</b> | <b>0.0956 <math>\pm</math> 0.0011</b> |
|             | $ADAM_U$  | 47     | 0.2108 $\pm$ 0.0036                   | 0.1442 $\pm$ 0.0027                   | 0.0978 $\pm$ 0.0016                   |
|             | EO        | 34     | 0.2402 $\pm$ 0.0047                   | 0.1773 $\pm$ 0.0053                   | 0.1114 $\pm$ 0.0022                   |
|             | EO (M)    | 4      | 0.2199 $\pm$ 0.0038                   | 0.1559 $\pm$ 0.0046                   | 0.1020 $\pm$ 0.0018                   |
|             | MPA       | 2      | 0.3922 $\pm$ 0.2004                   | 0.2984 $\pm$ 0.1493                   | 0.1819 $\pm$ 0.0929                   |
|             | MPA (M)   | 17     | 0.3976 $\pm$ 0.2313                   | 0.2903 $\pm$ 0.1711                   | 0.1844 $\pm$ 0.1073                   |
|             | PO        | 13     | 0.2675 $\pm$ 0.0049                   | 0.2034 $\pm$ 0.0042                   | 0.1241 $\pm$ 0.0023                   |
|             | PO (M)    | 20     | 0.2541 $\pm$ 0.0047                   | 0.1878 $\pm$ 0.0039                   | 0.1179 $\pm$ 0.0022                   |
|             | PSO       | 5      | 0.3813 $\pm$ 0.0522                   | 0.2860 $\pm$ 0.0304                   | 0.1769 $\pm$ 0.0242                   |
|             | PSO (M)   | 32     | 0.2791 $\pm$ 0.0274                   | 0.2113 $\pm$ 0.0247                   | 0.1294 $\pm$ 0.0127                   |
|             | WOA       | 1      | 0.2965 $\pm$ 0.0022                   | 0.2253 $\pm$ 0.0019                   | 0.1375 $\pm$ 0.0010                   |
|             | WOA (M)   | 74     | 0.2100 $\pm$ 0.0026                   | 0.1457 $\pm$ 0.0048                   | 0.0974 $\pm$ 0.0012                   |

Table 6. Cont.

| Time Series | Algorithm | Hidden | RMSE                   | MAE                    | NRMSE                  |
|-------------|-----------|--------|------------------------|------------------------|------------------------|
| Building 12 | $ADAM_T$  | 31     | <b>0.7166 ± 0.0056</b> | <b>0.5091 ± 0.0064</b> | <b>0.1755 ± 0.0014</b> |
|             | $ADAM_U$  | 32     | 0.7397 ± 0.0289        | 0.5253 ± 0.0250        | 0.1812 ± 0.0071        |
|             | EO        | 52     | 0.7854 ± 0.0074        | 0.5924 ± 0.0066        | 0.1924 ± 0.0018        |
|             | EO (M)    | 46     | 0.7580 ± 0.0040        | 0.5876 ± 0.0152        | 0.1856 ± 0.0010        |
|             | MPA       | 17     | 1.0764 ± 0.4315        | 0.8142 ± 0.3268        | 0.2636 ± 0.1057        |
|             | MPA (M)   | 10     | 1.0761 ± 0.3928        | 0.7926 ± 0.3075        | 0.2636 ± 0.0962        |
|             | PO        | 2      | 0.8166 ± 0.0139        | 0.6083 ± 0.0229        | 0.2000 ± 0.0034        |
|             | PO (M)    | 45     | 0.8166 ± 0.0135        | 0.6060 ± 0.0134        | 0.2000 ± 0.0033        |
|             | PSO       | 1      | 0.9230 ± 0.0290        | 0.7218 ± 0.0253        | 0.2261 ± 0.0071        |
|             | PSO (M)   | 14     | 0.9052 ± 0.0926        | 0.7021 ± 0.0849        | 0.2217 ± 0.0227        |
|             | WOA       | 1      | 0.9249 ± 0.0086        | 0.7636 ± 0.0048        | 0.2265 ± 0.0021        |
|             | WOA (M)   | 15     | 0.7405 ± 0.0122        | 0.5322 ± 0.0282        | 0.1814 ± 0.0030        |
| Building 14 | $ADAM_T$  | 27     | <b>2.2058 ± 0.0398</b> | <b>1.5337 ± 0.0232</b> | <b>0.1003 ± 0.0018</b> |
|             | $ADAM_U$  | 47     | 2.4099 ± 0.0260        | 1.7475 ± 0.0226        | 0.1095 ± 0.0012        |
|             | EO        | 31     | 2.5553 ± 0.0308        | 1.9211 ± 0.0436        | 0.1162 ± 0.0014        |
|             | EO (M)    | 5      | 2.2950 ± 0.0377        | 1.6398 ± 0.0334        | 0.1043 ± 0.0017        |
|             | MPA       | 9      | 5.0535 ± 2.6877        | 3.8798 ± 1.8926        | 0.2297 ± 0.1222        |
|             | MPA (M)   | 6      | 4.2389 ± 2.6920        | 3.0208 ± 1.9147        | 0.1927 ± 0.1224        |
|             | PO        | 14     | 2.8738 ± 0.0781        | 2.2206 ± 0.0778        | 0.1306 ± 0.0036        |
|             | PO (M)    | 45     | 2.5802 ± 0.1263        | 1.9215 ± 0.1148        | 0.1173 ± 0.0057        |
|             | PSO       | 10     | 4.6900 ± 0.3969        | 3.5742 ± 0.2505        | 0.2132 ± 0.0180        |
|             | PSO (M)   | 40     | 2.6126 ± 0.1489        | 1.9509 ± 0.1362        | 0.1188 ± 0.0068        |
|             | WOA       | 1      | 3.1706 ± 0.0914        | 2.5353 ± 0.0698        | 0.1441 ± 0.0042        |
|             | WOA (M)   | 48     | 2.2726 ± 0.1757        | 1.6352 ± 0.1775        | 0.1033 ± 0.0080        |
| Building 41 | $ADAM_T$  | 58     | 7.5326 ± 0.1444        | 5.8886 ± 0.1178        | 0.1846 ± 0.0035        |
|             | $ADAM_U$  | 80     | 7.6345 ± 0.3124        | 5.8330 ± 0.1691        | 0.1871 ± 0.0077        |
|             | EO        | 23     | 8.7176 ± 0.2243        | 6.4963 ± 0.2197        | 0.2136 ± 0.0055        |
|             | EO (M)    | 6      | 7.6711 ± 0.2987        | 5.7388 ± 0.1764        | 0.1880 ± 0.0073        |
|             | MPA       | 22     | 12.4294 ± 4.2551       | 9.5337 ± 3.3443        | 0.3046 ± 0.1043        |
|             | MPA (M)   | 16     | 12.1431 ± 5.5743       | 9.1791 ± 4.3850        | 0.2976 ± 0.1366        |
|             | PO        | 18     | 10.0713 ± 0.1242       | 7.5504 ± 0.0847        | 0.2468 ± 0.0030        |
|             | PO (M)    | 1      | 9.7443 ± 0.3678        | 7.3208 ± 0.2512        | 0.2388 ± 0.0090        |
|             | PSO       | 15     | 11.5497 ± 0.5485       | 8.8872 ± 0.3970        | 0.2830 ± 0.0134        |
|             | PSO (M)   | 26     | 11.0554 ± 2.2027       | 8.3852 ± 1.7870        | 0.2709 ± 0.0540        |
|             | WOA       | 55     | 12.9207 ± 1.0846       | 9.9794 ± 0.8659        | 0.3166 ± 0.0266        |
|             | WOA (M)   | 72     | <b>6.7206 ± 0.1299</b> | <b>5.1224 ± 0.1100</b> | <b>0.1647 ± 0.0032</b> |
| Building 57 | $ADAM_T$  | 62     | <b>8.0702 ± 0.1589</b> | <b>5.9193 ± 0.2068</b> | <b>0.2037 ± 0.0040</b> |
|             | $ADAM_U$  | 83     | 8.1546 ± 0.0740        | 6.1809 ± 0.1627        | 0.2058 ± 0.0019        |
|             | EO        | 5      | 8.5308 ± 0.1852        | 6.6911 ± 0.1495        | 0.2153 ± 0.0047        |
|             | EO (M)    | 86     | 8.7038 ± 0.2799        | 6.8639 ± 0.2919        | 0.2197 ± 0.0071        |
|             | MPA       | 1      | 10.5879 ± 1.9252       | 7.9185 ± 1.2021        | 0.2672 ± 0.0486        |
|             | MPA (M)   | 57     | 13.4660 ± 6.7047       | 10.4599 ± 5.0262       | 0.3398 ± 0.1692        |
|             | PO        | 14     | 8.5193 ± 0.1302        | 6.6637 ± 0.0704        | 0.2150 ± 0.0033        |
|             | PO (M)    | 7      | 8.5655 ± 0.3661        | 6.7089 ± 0.2881        | 0.2162 ± 0.0092        |
|             | PSO       | 4      | 10.4601 ± 1.4624       | 7.9638 ± 0.8150        | 0.2640 ± 0.0369        |
|             | PSO (M)   | 11     | 9.6639 ± 1.2333        | 7.4983 ± 0.9644        | 0.2439 ± 0.0311        |
|             | WOA       | 80     | 11.2813 ± 0.7236       | 8.3118 ± 0.7372        | 0.2847 ± 0.0183        |
|             | WOA (M)   | 22     | 8.3449 ± 0.1749        | 6.5695 ± 0.1660        | 0.2106 ± 0.0044        |
| Building 60 | $ADAM_T$  | 77     | <b>2.9764 ± 0.0307</b> | <b>1.7581 ± 0.0118</b> | <b>0.0850 ± 0.0009</b> |
|             | $ADAM_U$  | 49     | 3.1370 ± 0.0806        | 1.9672 ± 0.1307        | 0.0896 ± 0.0023        |
|             | EO        | 54     | 3.5171 ± 0.1091        | 2.3498 ± 0.0761        | 0.1005 ± 0.0031        |
|             | EO (M)    | 4      | 3.2018 ± 0.0198        | 2.1072 ± 0.0176        | 0.0915 ± 0.0006        |
|             | MPA       | 11     | 6.1387 ± 3.9623        | 4.5685 ± 3.2480        | 0.1754 ± 0.1132        |
|             | MPA (M)   | 9      | 5.6353 ± 3.3208        | 3.7708 ± 2.2881        | 0.1610 ± 0.0949        |
|             | PO        | 7      | 3.9626 ± 0.1317        | 2.6050 ± 0.0776        | 0.1132 ± 0.0038        |
|             | PO (M)    | 43     | 3.7901 ± 0.1649        | 2.5118 ± 0.1202        | 0.1083 ± 0.0047        |
|             | PSO       | 14     | 6.3096 ± 0.4395        | 4.6375 ± 0.3908        | 0.1803 ± 0.0126        |
|             | PSO (M)   | 27     | 4.9914 ± 2.0579        | 3.5182 ± 1.7033        | 0.1426 ± 0.0588        |
|             | WOA       | 31     | 4.7560 ± 1.1289        | 3.3499 ± 1.0379        | 0.1359 ± 0.0323        |
|             | WOA (M)   | 17     | 3.1493 ± 0.0520        | 1.8942 ± 0.0691        | 0.0900 ± 0.0015        |

Table 6. Cont.

| Time Series | Algorithm | Hidden | RMSE                                   | MAE                                    | NRMSE                                 |
|-------------|-----------|--------|----------------------------------------|----------------------------------------|---------------------------------------|
| Uruguay     | $ADAM_T$  | 14     | $103.5332 \pm 9.9878$                  | $75.9769 \pm 5.8770$                   | $0.0835 \pm 0.0081$                   |
|             | $ADAM_U$  | 81     | $102.0505 \pm 22.3021$                 | $76.8430 \pm 20.2218$                  | $0.0823 \pm 0.0180$                   |
|             | EO        | 2      | $109.1516 \pm 3.4165$                  | $83.7064 \pm 2.0885$                   | $0.0880 \pm 0.0028$                   |
|             | EO (M)    | 2      | $102.2170 \pm 4.6378$                  | $78.1375 \pm 3.7598$                   | $0.0824 \pm 0.0037$                   |
|             | MPA       | 15     | $185.4720 \pm 78.2519$                 | $141.3799 \pm 57.3471$                 | $0.1496 \pm 0.0631$                   |
|             | MPA (M)   | 2      | $163.8886 \pm 71.7491$                 | $127.3167 \pm 56.5189$                 | $0.1322 \pm 0.0579$                   |
|             | PO        | 1      | $104.1256 \pm 2.0937$                  | $79.7465 \pm 1.5264$                   | $0.0840 \pm 0.0017$                   |
|             | PO (M)    | 2      | $103.6811 \pm 4.5076$                  | $79.1997 \pm 3.8771$                   | $0.0836 \pm 0.0036$                   |
|             | PSO       | 4      | $167.9046 \pm 12.2412$                 | $129.1816 \pm 10.1764$                 | $0.1354 \pm 0.0099$                   |
|             | PSO (M)   | 12     | $110.8241 \pm 5.8375$                  | $84.9766 \pm 4.2414$                   | $0.0894 \pm 0.0047$                   |
|             | WOA       | 20     | $208.3396 \pm 8.9959$                  | $168.1214 \pm 6.3930$                  | $0.1680 \pm 0.0073$                   |
|             | WOA (M)   | 14     | <b><math>93.5938 \pm 1.3272</math></b> | <b><math>71.4459 \pm 1.4957</math></b> | <b><math>0.0755 \pm 0.0011</math></b> |

Note: Bold values indicate the best results for each metric and time series.

Having analyzed the forecasting error of the models, we now turn to their computational efficiency. Note that our primary goal is to develop accurate models quickly in order to obtain the best possible solutions within a short period of time. Thus, Table 7 presents a final comparison showing the best model from the best baseline and the best metaheuristic/memetic algorithm in each building. This table also includes the hidden size, training time, and results of the models trained with  $ADAM_U$ , allowing us to compare the complexity and training time with all approaches. In five instances,  $ADAM$  outperformed the best metaheuristic in terms of speed: Buildings 2, 9, 10, 41 and the Uruguay dataset, with an average speedup of approximately 80% in those cases. This advantage is mainly explained by the selected architecture, since the best  $ADAM$  model in these cases is generally an MLP trained with  $ADAM_U$ , whereas the best metaheuristic/memetic alternative is often an LSTM model. However, it is worth noting that the metaheuristic achieved a lower RMSE for Buildings 9 and 41, despite the higher computational cost associated with the selected LSTM architecture. On the other hand, when the best metaheuristic/memetic model was based on a simpler MLP architecture, the speedup was substantial: this occurs clearly in Buildings 1 and 12, where the proposed approach was more than 200 times and nearly 94 times faster, respectively. On average, the metaheuristic was approximately 56% faster when it outperformed  $ADAM$  in terms of execution time. Overall, these results show that the observed speedup is strongly linked to the architecture selected as the best model: MLP-based solutions tend to provide the largest computational gains, whereas LSTM-based solutions can improve accuracy in some cases at the cost of longer training times.

**Table 7.** Final comparison between the best ADAM baseline and the best metaheuristic/memetic model for each time series.

| Time Series | Model                            | Arch. | Hidden | RMSE                                   | MAE                                    | NRMSE                                 | Time (ms) |
|-------------|----------------------------------|-------|--------|----------------------------------------|----------------------------------------|---------------------------------------|-----------|
| Building 1  | <b>ADAM: <math>ADAM_T</math></b> | LSTM  | 86     | <b>0.1137 <math>\pm</math> 0.0009</b>  | <b>0.0782 <math>\pm</math> 0.0021</b>  | <b>0.1418 <math>\pm</math> 0.0011</b> | 34,541    |
|             | MHNN: EO (M)                     | MLP   | 7      | 0.1303 $\pm$ 0.0020                    | 0.1018 $\pm$ 0.0030                    | 0.1624 $\pm$ 0.0025                   | 159       |
| Building 2  | <b>ADAM: <math>ADAM_U</math></b> | MLP   | 86     | <b>0.2495 <math>\pm</math> 0.0055</b>  | <b>0.1994 <math>\pm</math> 0.0041</b>  | <b>0.2735 <math>\pm</math> 0.0060</b> | 242       |
|             | MHNN: WOA (M)                    | LSTM  | 42     | 0.2579 $\pm$ 0.0101                    | 0.2139 $\pm$ 0.0057                    | 0.2828 $\pm$ 0.0111                   | 25,968    |
| Building 3  | <b>ADAM: <math>ADAM_T</math></b> | LSTM  | 73     | <b>0.2631 <math>\pm</math> 0.0048</b>  | <b>0.1935 <math>\pm</math> 0.0047</b>  | <b>0.1639 <math>\pm</math> 0.0030</b> | 28,811    |
|             | MHNN: WOA (M)                    | LSTM  | 14     | 0.2704 $\pm$ 0.0034                    | 0.1956 $\pm$ 0.0053                    | 0.1685 $\pm$ 0.0021                   | 14,902    |
| Building 9  | <b>ADAM: <math>ADAM_U</math></b> | MLP   | 4      | 1.1237 $\pm$ 0.0082                    | <b>0.8072 <math>\pm</math> 0.0069</b>  | 0.2166 $\pm$ 0.0016                   | 175       |
|             | <b>MHNN: WOA (M)</b>             | LSTM  | 38     | <b>1.1136 <math>\pm</math> 0.0070</b>  | 0.8134 $\pm$ 0.0158                    | <b>0.2146 <math>\pm</math> 0.0013</b> | 36,572    |
| Building 10 | <b>ADAM: <math>ADAM_U</math></b> | MLP   | 66     | <b>0.2007 <math>\pm</math> 0.0030</b>  | <b>0.1371 <math>\pm</math> 0.0024</b>  | <b>0.0931 <math>\pm</math> 0.0014</b> | 278       |
|             | MHNN: WOA (M)                    | MLP   | 22     | 0.2063 $\pm$ 0.0043                    | 0.1453 $\pm$ 0.0053                    | 0.0957 $\pm$ 0.0020                   | 286       |
| Building 12 | <b>ADAM: <math>ADAM_T</math></b> | LSTM  | 31     | <b>0.7166 <math>\pm</math> 0.0056</b>  | <b>0.5091 <math>\pm</math> 0.0064</b>  | <b>0.1755 <math>\pm</math> 0.0014</b> | 18,235    |
|             | MHNN: EO (M)                     | MLP   | 6      | 0.7338 $\pm$ 0.0067                    | 0.5268 $\pm$ 0.0124                    | 0.1797 $\pm$ 0.0016                   | 193       |
| Building 14 | <b>ADAM: <math>ADAM_U</math></b> | MLP   | 90     | <b>2.1826 <math>\pm</math> 0.0181</b>  | <b>1.5399 <math>\pm</math> 0.0241</b>  | <b>0.0992 <math>\pm</math> 0.0008</b> | 302       |
|             | MHNN: EO (M)                     | MLP   | 5      | 2.2620 $\pm$ 0.0258                    | 1.6280 $\pm$ 0.0384                    | 0.1028 $\pm$ 0.0012                   | 190       |
| Building 41 | <b>ADAM: <math>ADAM_U</math></b> | MLP   | 80     | 6.9385 $\pm$ 0.0959                    | 5.4881 $\pm$ 0.0544                    | 0.1700 $\pm$ 0.0023                   | 214       |
|             | <b>MHNN: WOA (M)</b>             | LSTM  | 72     | <b>6.7206 <math>\pm</math> 0.1299</b>  | <b>5.1224 <math>\pm</math> 0.1100</b>  | <b>0.1647 <math>\pm</math> 0.0032</b> | 40,559    |
| Building 57 | <b>ADAM: <math>ADAM_U</math></b> | MLP   | 79     | <b>7.4302 <math>\pm</math> 0.0432</b>  | <b>5.5718 <math>\pm</math> 0.0518</b>  | <b>0.1875 <math>\pm</math> 0.0011</b> | 184       |
|             | MHNN: WOA (M)                    | MLP   | 83     | 7.7682 $\pm$ 0.2200                    | 5.8580 $\pm$ 0.2308                    | 0.1960 $\pm$ 0.0056                   | 178       |
| Building 60 | <b>ADAM: <math>ADAM_T</math></b> | LSTM  | 77     | <b>2.9764 <math>\pm</math> 0.0307</b>  | <b>1.7581 <math>\pm</math> 0.0118</b>  | <b>0.0850 <math>\pm</math> 0.0009</b> | 32,192    |
|             | MHNN: WOA (M)                    | LSTM  | 17     | 3.1493 $\pm$ 0.0520                    | 1.8942 $\pm$ 0.0691                    | 0.0900 $\pm$ 0.0015                   | 16,751    |
| Uruguay     | <b>ADAM: <math>ADAM_U</math></b> | MLP   | 44     | <b>78.0887 <math>\pm</math> 7.0743</b> | <b>57.0153 <math>\pm</math> 6.2761</b> | <b>0.0630 <math>\pm</math> 0.0057</b> | 840       |
|             | MHNN: WOA (M)                    | LSTM  | 14     | 93.5938 $\pm$ 1.3272                   | 71.4459 $\pm$ 1.4957                   | 0.0755 $\pm$ 0.0011                   | 72,573    |

Note: For each time series, bold metric values indicate the lowest RMSE, MAE, or NRMSE. Bold model names indicate the model that achieves the lowest values in the majority of these error metrics.

The descriptive results therefore point to several related issues: Whether the memetic component systematically improves the corresponding base metaheuristic, whether the hidden-layer size affects the performance of the proposed metaheuristic and memetic training methods, and whether these methods outperform  $ADAM_T$  under matched experimental configurations. To analyse these aspects more formally, paired Wilcoxon signed-rank tests were applied.

The memetic-vs-base tests displayed in Table 8 show significant improvements for EO (M), PSO (M) and WOA (M) in MLP, and for PSO (M) and WOA (M) in LSTM. No significant improvement is observed for ENN, and MPA does not show a significant gain in any architecture. These results are consistent with the descriptive analysis of the tables. The memetic component is clearly beneficial in MLP and LSTM, especially for WOA (M), but it should not be interpreted as a uniformly advantageous addition to every population-based optimizer or every neural architecture.

**Table 8.** Wilcoxon tests comparing memetic variants against their base metaheuristics.

| Architecture | Base | Memetic | Mean Diff | $p$       | Holm $p$ |
|--------------|------|---------|-----------|-----------|----------|
| ENN          | EO   | EO (M)  | 14.8512   | 1         | 1        |
| ENN          | MPA  | MPA (M) | −0.9894   | 0.1201    | 0.8408   |
| ENN          | PO   | PO (M)  | 0.3357    | 0.9492    | 1        |
| ENN          | PSO  | PSO (M) | 20.4996   | 1         | 1        |
| ENN          | WOA  | WOA (M) | 18.8532   | 0.0874    | 0.6992   |
| LSTM         | EO   | EO (M)  | −0.7708   | 0.01221   | 0.1099   |
| LSTM         | MPA  | MPA (M) | −1.8474   | 0.1602    | 0.8408   |
| LSTM         | PO   | PO (M)  | −0.1134   | 0.009277  | 0.09277  |
| LSTM         | PSO  | PSO (M) | −5.6450   | 0.0004883 | 0.007324 |
| LSTM         | WOA  | WOA (M) | −11.5896  | 0.0004883 | 0.007324 |
| MLP          | EO   | EO (M)  | −1.3604   | 0.0004883 | 0.007324 |
| MLP          | MPA  | MPA (M) | −3.5066   | 0.7676    | 1        |
| MLP          | PO   | PO (M)  | −1.1539   | 0.1392    | 0.8408   |
| MLP          | PSO  | PSO (M) | −6.0146   | 0.0009766 | 0.01074  |
| MLP          | WOA  | WOA (M) | −12.0021  | 0.0004883 | 0.007324 |

Table 9 examines the influence of the hidden layer size on metaheuristic and memetic training performance. For each architecture, configurations from the lower part of the

evaluated hidden neuron range were compared with configurations from the upper part of the range in terms of RMSE. This comparison was used to assess whether these methods perform differently in smaller and larger neural search spaces. The results indicate that configurations with fewer hidden neurons tend to be more favorable for ENN and, to a lesser extent, for MLP. In contrast, no significant difference is observed for LSTM. This behavior may be related to the different number of trainable parameters associated with each architecture. Although increasing the number of hidden neurons enlarges the search space in all models, LSTM networks are already considerably more parameterized because of their gate-specific transformations and recurrent connections. As a result, even LSTM configurations with fewer hidden neurons may represent a demanding search space for the fixed evaluation budget, making the difference between both groups less pronounced.

**Table 9.** Wilcoxon tests comparing configurations with 1–45 and 46–90 hidden neurons within each neural architecture.

| Architecture | 1–45 Neurons | 46–90 Neurons | W   | p                      |
|--------------|--------------|---------------|-----|------------------------|
| ENN          | 102.4964     | 594.7111      | 4   | $1.669 \times 10^{-6}$ |
| LSTM         | 73.7934      | 76.5806       | 60  | 0.0844                 |
| MLP          | 51.8384      | 66.0893       | 171 | 0.04186                |

Table 10 reports the matched comparison between  $ADAM_T$  and each proposed metaheuristic or memetic method. Each pair corresponds to the same dataset, architecture, hidden neuron count, and seed. The overall rows aggregate the three neural architectures, whereas the remaining rows report the same comparison separately for MLP, ENN, and LSTM. The Wins column reports the number of matched cases in which the proposed method obtains a lower RMSE than  $ADAM_T$ , and negative mean differences indicate a lower RMSE for the proposed method. The matched configuration tests provide evidence in favour of the proposed metaheuristic and memetic methods when  $ADAM_T$  is restricted by the same training time. When the dataset, architecture, hidden neuron count, and seed are fixed, several algorithms significantly outperform  $ADAM_T$ , particularly for MLP configurations and for ENN comparisons involving PO. However, this pattern is not observed for LSTM, where ADAM remains highly competitive. This apparent contrast with the final comparison is expected: the matched tests evaluate identical configurations against  $ADAM_T$ , whereas Table 7 compares the best available model from each family. Thus, the statistical tests show that the proposed training strategies can be preferable under constrained matched settings, while the final table shows that unconstrained or selected ADAM remains stronger in most series.

Overall, the results indicate that the proposed CUDA implementation of metaheuristic and memetic training strategies is mainly useful as a competitive alternative under strict training budgets, rather than as a method that systematically dominates ADAM. Its most convincing behavior appears in LSTM models trained with WOA (M), while ENN results are less robust and MLP results depend strongly on whether the comparison is made against time-limited or unconstrained ADAM. This is relevant in building-energy applications where models may need to be retrained frequently or adapted quickly after changes in consumption behavior, since in those scenarios a competitive method with favorable execution characteristics can still be valuable even when ADAM remains the strongest overall baseline at the expense of a higher computational budget.

**Table 10.** Matched-configuration Wilcoxon tests comparing  $ADAM_T$  against MHNN algorithms. Overall rows aggregate MLP, ENN, and LSTM configurations, while architecture-specific rows restrict the comparison to a single architecture.

| Baseline | Arch.   | Algorithm | Wins | Mean Diff. | $p$                      | Holm $p$                 |
|----------|---------|-----------|------|------------|--------------------------|--------------------------|
| $ADAM_T$ | Overall | EO        | 112  | −1.6014    | $4.282 \times 10^{-9}$   | $1.328 \times 10^{-7}$   |
| $ADAM_T$ | Overall | EO (M)    | 16   | 3.1575     | 0.6744                   | 1                        |
| $ADAM_T$ | Overall | MPA       | 103  | −3.1256    | $7.264 \times 10^{-17}$  | $2.397 \times 10^{-15}$  |
| $ADAM_T$ | Overall | MPA (M)   | 106  | −0.0826    | $3.071 \times 10^{-14}$  | $9.826 \times 10^{-13}$  |
| $ADAM_T$ | Overall | PO        | 23   | −5.8950    | 0.001578                 | 0.03472                  |
| $ADAM_T$ | Overall | PO (M)    | 23   | −5.8063    | 0.0002167                | 0.006285                 |
| $ADAM_T$ | Overall | PSO       | 18   | −1.2432    | 0.2                      | 1                        |
| $ADAM_T$ | Overall | PSO (M)   | 444  | −3.2061    | $9.776 \times 10^{-67}$  | $3.519 \times 10^{-65}$  |
| $ADAM_T$ | Overall | WOA       | 553  | −5.0359    | $1.176 \times 10^{-89}$  | $4.469 \times 10^{-88}$  |
| $ADAM_T$ | Overall | WOA (M)   | 1133 | 12.8485    | 1                        | 1                        |
| $ADAM_T$ | ENN     | EO        | 101  | −1.6460    | $6.83 \times 10^{-9}$    | $2.049 \times 10^{-7}$   |
| $ADAM_T$ | ENN     | EO (M)    | 0    | 12.0967    | 1                        | 1                        |
| $ADAM_T$ | ENN     | MPA       | 4    | −6.4221    | 0.7402                   | 1                        |
| $ADAM_T$ | ENN     | MPA (M)   | 7    | −7.5369    | 0.07373                  | 1                        |
| $ADAM_T$ | ENN     | PO        | 11   | −16.5842   | 0.0004883                | 0.01367                  |
| $ADAM_T$ | ENN     | PO (M)    | 11   | −13.9546   | 0.0004883                | 0.01367                  |
| $ADAM_T$ | ENN     | PSO       | 9    | −10.3038   | 0.009277                 | 0.1948                   |
| $ADAM_T$ | ENN     | PSO (M)   | 1    | 9.8497     | 0.9976                   | 1                        |
| $ADAM_T$ | ENN     | WOA       | 8    | −5.5271    | 0.1602                   | 1                        |
| $ADAM_T$ | ENN     | WOA (M)   | 24   | 41.2461    | 1                        | 1                        |
| $ADAM_T$ | LSTM    | EO        | 1    | 1.9022     | 0.9951                   | 1                        |
| $ADAM_T$ | LSTM    | EO (M)    | 6    | 0.7193     | 0.4155                   | 1                        |
| $ADAM_T$ | LSTM    | MPA       | 0    | 10.4663    | 1                        | 1                        |
| $ADAM_T$ | LSTM    | MPA (M)   | 0    | 11.9321    | 1                        | 1                        |
| $ADAM_T$ | LSTM    | PO        | 2    | 2.1441     | 0.9932                   | 1                        |
| $ADAM_T$ | LSTM    | PO (M)    | 2    | 0.4781     | 0.9492                   | 1                        |
| $ADAM_T$ | LSTM    | PSO       | 0    | 7.4659     | 1                        | 1                        |
| $ADAM_T$ | LSTM    | PSO (M)   | 0    | 1.3116     | 1                        | 1                        |
| $ADAM_T$ | LSTM    | WOA       | 1    | 11.4898    | 0.9995                   | 1                        |
| $ADAM_T$ | LSTM    | WOA (M)   | 299  | 0.0847     | 1                        | 1                        |
| $ADAM_T$ | MLP     | EO        | 10   | −4.6119    | 0.0009766                | 0.02539                  |
| $ADAM_T$ | MLP     | EO (M)    | 10   | −3.9936    | 0.0009766                | 0.02539                  |
| $ADAM_T$ | MLP     | MPA       | 99   | −4.2696    | $2.849 \times 10^{-18}$  | $9.97 \times 10^{-17}$   |
| $ADAM_T$ | MLP     | MPA (M)   | 99   | −0.5893    | $2.849 \times 10^{-18}$  | $9.97 \times 10^{-17}$   |
| $ADAM_T$ | MLP     | PO        | 10   | −2.9800    | 0.0009766                | 0.02539                  |
| $ADAM_T$ | MLP     | PO (M)    | 10   | −3.7561    | 0.0009766                | 0.02539                  |
| $ADAM_T$ | MLP     | PSO       | 9    | −0.8566    | 0.01855                  | 0.3711                   |
| $ADAM_T$ | MLP     | PSO (M)   | 443  | −3.6310    | $1.359 \times 10^{-71}$  | $5.027 \times 10^{-70}$  |
| $ADAM_T$ | MLP     | WOA       | 544  | −5.3602    | $4.139 \times 10^{-91}$  | $1.614 \times 10^{-89}$  |
| $ADAM_T$ | MLP     | WOA (M)   | 810  | −5.9209    | $1.118 \times 10^{-134}$ | $4.472 \times 10^{-133}$ |

## 6. Conclusions

The present study evaluated GPU-accelerated metaheuristic and memetic algorithms for training the weights and biases of neural forecasting models in short-term energy-consumption prediction. To this end, efficient CUDA implementations of three ANN architectures, namely MLP, ENN, and LSTM, were developed together with five population-based metaheuristic optimizers and their corresponding memetic variants using an  $ADAM$ -based local search. These methods were compared with TensorFlow GPU baselines on 11 real-world energy-consumption time series, considering training time and forecasting accuracy through RMSE, MAE, and NRMSE.

The results show that the proposed CUDA framework for metaheuristic and memetic neural-network training can provide a competitive alternative to conventional gradient-based training when the available training time is strongly constrained. This behavior is especially relevant in building-energy management scenarios where forecasting models may

need to be retrained frequently or rapidly adapted after changes in consumption patterns. Under matched time-budgeted settings, several metaheuristic and memetic configurations improve upon the time-constrained  $ADAM_T$  baseline, particularly for MLP models and selected recurrent configurations. However, the results also show that unconstrained or best-selected ADAM remains a strong overall reference and obtains the best results in most time series when the training budget is not restricted. Therefore, the proposed approach should be interpreted as a rapid-training alternative for time-sensitive forecasting scenarios rather than as a universally superior replacement for gradient-based optimization.

The behavior of the different neural architectures also reveals relevant differences. For MLP models, memetic variants generally provide the most favorable trade-off between accuracy and computational cost under restricted training budgets. For ENNs, the results are less robust, likely because the local-search component can be affected by vanishing or exploding gradients during backpropagation through time. In contrast, LSTM models benefit more consistently from the memetic strategy, probably because their gated structure helps mitigate these recurrent-training issues. Among the evaluated optimizers, WOA (M) and EO emerged as the most competitive alternatives, although no single method dominated across all datasets and architectures.

Future work will focus on extending the CUDA framework with additional metaheuristic and hybrid optimization strategies, improving the behavior of memetic algorithms in larger recurrent search spaces, and studying adaptive mechanisms to select the most suitable optimizer depending on the dataset, architecture, and available training budget.

**Author Contributions:** D.C.-R.: Conceptualization, Methodology, Software, Investigation, Writing—original draft; M.R.C.: Software, Investigation, Writing—review & editing; L.G.B.R.: Methodology, Investigation, Writing—original draft, Writing—review & editing; L.S.: Methodology, Validation, Supervision, Writing—review & editing; R.W.: Methodology, Validation, Supervision, Writing—review & editing; M.P.C.: Methodology, Validation, Resources, Writing—review & editing; M.C.P.: Conceptualization, Methodology, Resources, Supervision, Writing—review & editing. All authors have read and agreed to the published version of the manuscript.

**Funding:** We acknowledge financial support from Ministerio de Ciencia e Innovación (Spain) (Grant PID2020-112495RB-C21 funded by MCIN/AEI/10.13039/501100011033).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The original data presented in the study are openly available in Github at <https://github.com/CDAC-lab/UNICON> (accessed on 5 September 2025). The Uruguay energy consumption data are available from the corresponding author upon reasonable request.

**Acknowledgments:** The authors acknowledge the providers of the datasets used in this study.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

|       |                                          |
|-------|------------------------------------------|
| ABC   | Artificial bee colony                    |
| ADAM  | Adaptive moment estimation               |
| AI    | Artificial intelligence                  |
| ANN   | Artificial neural network                |
| ARIMA | Autoregressive integrated moving average |
| CPU   | Central processing unit                  |
| CUDA  | Compute Unified Device Architecture      |
| ENN   | Elman neural network                     |

|         |                                            |
|---------|--------------------------------------------|
| EO      | Equilibrium Optimizer                      |
| FP32    | 32-bit floating-point precision            |
| GA      | Genetic algorithm                          |
| GPU     | Graphics processing unit                   |
| GRU     | Gated recurrent unit                       |
| LSTM    | Long short-term memory                     |
| MAE     | Mean absolute error                        |
| MLP     | Multi-layer perceptron                     |
| MPA     | Marine Predators Algorithm                 |
| NRMSE   | Normalized root mean squared error         |
| NSGA-II | Non-dominated Sorting Genetic Algorithm II |
| PCIe    | Peripheral Component Interconnect Express  |
| PO      | Political Optimizer                        |
| PSO     | Particle Swarm Optimization                |
| RMSE    | Root mean squared error                    |
| RNN     | Recurrent neural network                   |
| SA      | Simulated annealing                        |
| SIMT    | Single Instruction Multiple Threads        |
| WOA     | Whale Optimization Algorithm               |

## References

1. Safarzadeh, S.; Rasti-Barzoki, M.; Hejazi, S.R. A review of optimal energy policy instruments on industrial energy efficiency programs, rebound effects, and government policies. *Energy Policy* **2020**, *139*, 111342. [\[CrossRef\]](#)
2. Dehler, J.; Keles, D.; Telsnig, T.; Fleischer, B.; Baumann, M.; Fraboulet, D.; Faure-Schuyer, A.; Fichtner, W. Chapter 27—Self-Consumption of Electricity from Renewable Sources. In *Europe's Energy Transition*; Welsch, M., Pye, S., Keles, D., Faure-Schuyer, A., Dobbins, A., Shivakumar, A., Deane, P., Howells, M., Eds.; Academic Press: Cambridge, MA, USA, 2017; pp. 225–236. [\[CrossRef\]](#)
3. O'Connell, N.; Pinson, P.; Madsen, H.; O'Malley, M. Benefits and challenges of electrical demand response: A critical review. *Renew. Sustain. Energy Rev.* **2014**, *39*, 686–699. [\[CrossRef\]](#)
4. Al-Ogaili, A.S.; Tengku Hashim, T.J.; Rahmat, N.A.; Ramasamy, A.K.; Marsadek, M.B.; Faisal, M.; Hannan, M.A. Review on Scheduling, Clustering, and Forecasting Strategies for Controlling Electric Vehicle Charging: Challenges and Recommendations. *IEEE Access* **2019**, *7*, 128353–128371. [\[CrossRef\]](#)
5. Frieden, D.; Tuerk, A.; Neumann, C.; d'Herbemont, S.; Roberts, J. Collective Self-Consumption and Energy Communities: Trends and Challenges in the Transposition of the EU Framework. *Working Paper*; COMPILE Project; 2020. Available online: <https://www.rescoop.eu/toolbox/collective-self-consumption-and-energy-communities-trends-and-challenges-in-the-transposition-of-the-eu-framework> (accessed on 19 May 2026).
6. Shaikh, I. Impact of COVID-19 pandemic on the energy markets. *Econ. Change Restruct.* **2022**, *55*, 433–484.
7. Taylor, J.W. Triple seasonal methods for short-term electricity demand forecasting. *Eur. J. Oper. Res.* **2010**, *204*, 139–152. [\[CrossRef\]](#)
8. Nepal, B.; Yamaha, M.; Yokoe, A.; Yamaji, T. Electricity load forecasting using clustering and ARIMA model for energy management in buildings. *Jpn. Archit. Rev.* **2020**, *3*, 62–76.
9. Wei, Y.; Zhang, X.; Shi, Y.; Xia, L.; Pan, S.; Wu, J.; Han, M.; Zhao, X. A review of data-driven approaches for prediction and classification of building energy consumption. *Renew. Sustain. Energy Rev.* **2018**, *82*, 1027–1047. [\[CrossRef\]](#)
10. Wang, J.Q.; Du, Y.; Wang, J. LSTM based long-term energy consumption prediction with periodicity. *Energy* **2020**, *197*, 117197. [\[CrossRef\]](#)
11. Michailidis, P.; Michailidis, I.; Gkelios, S.; Kosmatopoulos, E. Artificial Neural Network Applications for Energy Management in Buildings: Current Trends and Future Directions. *Energies* **2024**, *17*, 570. [\[CrossRef\]](#)
12. Massaoudi, M.; Refaat, S.S.; Chihi, I.; Trabelsi, M.; Oueslati, F.S.; Abu-Rub, H. A novel stacked generalization ensemble-based hybrid LGBM-XGB-MLP model for Short-Term Load Forecasting. *Energy* **2021**, *214*, 118874. [\[CrossRef\]](#)
13. Manno, A.; Martelli, E.; Amaldi, E. A Shallow Neural Network Approach for the Short-Term Forecast of Hourly Energy Consumption. *Energies* **2022**, *15*, 958. [\[CrossRef\]](#)
14. Criado-Ramón, D.; Ruiz, L.; Pegalajar, M. Electric demand forecasting with neural networks and symbolic time series representations. *Appl. Soft Comput.* **2022**, *122*, 108871. [\[CrossRef\]](#)
15. Blanco, A.; Delgado, M.; Pegalajar, M. A genetic algorithm to obtain the optimal recurrent neural network. *Int. J. Approx. Reason.* **2000**, *23*, 67–83. [\[CrossRef\]](#)

16. Blanco, A.; Delgado, M.; Pegalajar, M. A real-coded genetic algorithm for training recurrent neural networks. *Neural Netw.* **2001**, *14*, 93–105. [[CrossRef](#)] [[PubMed](#)]
17. Bacanin, N.; Jovanovic, L.; Zivkovic, M.; Kandasamy, V.; Antonijevic, M.; Deveci, M.; Strumberger, I. Multivariate energy forecasting via metaheuristic tuned long-short term memory and gated recurrent unit neural networks. *Inf. Sci.* **2023**, *642*, 119122. [[CrossRef](#)]
18. Muralitharan, K.; Sakthivel, R.; Vishnuvarthan, R. Neural network based optimization approach for energy demand prediction in smart grid. *Neurocomputing* **2018**, *273*, 199–208. [[CrossRef](#)]
19. Zhou, G.; Moayed, H.; Bahiraei, M.; Lyu, Z. Employing artificial bee colony and particle swarm techniques for optimizing a neural network in prediction of heating and cooling loads of residential buildings. *J. Clean. Prod.* **2020**, *254*, 120082. [[CrossRef](#)]
20. Bouktif, S.; Fiaz, A.; Ouni, A.; Serhani, M.A. Multi-Sequence LSTM-RNN Deep Learning and Metaheuristics for Electric Load Forecasting. *Energies* **2020**, *13*, 391. [[CrossRef](#)]
21. Luo, X.; Oyedele, L.O.; Ajayi, A.O.; Akinade, O.O.; Owolabi, H.A.; Ahmed, A. Feature extraction and genetic algorithm enhanced adaptive deep neural network for energy consumption prediction in buildings. *Renew. Sustain. Energy Rev.* **2020**, *131*, 109980. [[CrossRef](#)]
22. Luo, X.; Oyedele, L.O. Forecasting building energy consumption: Adaptive long-short term memory neural networks driven by genetic algorithm. *Adv. Eng. Inform.* **2021**, *50*, 101357. [[CrossRef](#)]
23. Phatai, G.; Chiewchanwattana, S.; Sunat, K. A Comparative of Neural Network with Metaheuristics for Electricity Consumption Forecast Modelling. In Proceedings of the 2018 22nd International Computer Science and Engineering Conference (ICSEC), Chiang Mai, Thailand, 21–24 November 2018; pp. 1–4. [[CrossRef](#)]
24. Sahraei, M.A.; Çodur, M.K. Prediction of transportation energy demand by novel hybrid meta-heuristic ANN. *Energy* **2022**, *249*, 123735. [[CrossRef](#)]
25. Ruiz, L.; Rueda, R.; Cuéllar, M.; Pegalajar, M. Energy consumption forecasting based on Elman neural networks with evolutive optimization. *Expert Syst. Appl.* **2018**, *92*, 380–389. [[CrossRef](#)]
26. Iruela, J.; Ruiz, L.; Pegalajar, M.; Capel, M. A parallel solution with GPU technology to predict energy consumption in spatially distributed buildings using evolutionary optimization and artificial neural networks. *Energy Convers. Manag.* **2020**, *207*, 112535. [[CrossRef](#)]
27. Iruela, J.; Ruiz, L.; Criado-Ramón, D.; Pegalajar, M.; Capel, M. A GPU-accelerated adaptation of the PSO algorithm for multi-objective optimization applied to artificial neural networks to predict energy consumption. *Appl. Soft Comput.* **2024**, *160*, 111711. [[CrossRef](#)]
28. Almeida, L.B. Multilayer perceptrons. In *Proceedings of the Handbook of Neural Computation*; IOP Publishing Ltd.: Oxford, UK; Oxford University Press: Oxford, UK, 1997.
29. Elman, J. Finding Structure in Time. *Cogn. Sci.* **1990**, *14*, 179–211. [[CrossRef](#)]
30. Hochreiter, S.; Schmidhuber, J. Long Short-term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)] [[PubMed](#)]
31. Hochreiter, S. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *Int. J. Uncertain. Fuzziness-Knowl.-Based Syst.* **1998**, *6*, 107–116. [[CrossRef](#)]
32. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948. [[CrossRef](#)]
33. Faramarzi, A.; Heidarinejad, M.; Stephens, B.; Mirjalili, S. Equilibrium optimizer: A novel optimization algorithm. *Knowl.-Based Syst.* **2020**, *191*, 105190. [[CrossRef](#)]
34. Faramarzi, A.; Heidarinejad, M.; Mirjalili, S.; Gandomi, A.H. Marine Predators Algorithm: A nature-inspired metaheuristic. *Expert Syst. Appl.* **2020**, *152*, 113377. [[CrossRef](#)]
35. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [[CrossRef](#)]
36. Askari, Q.; Younas, I.; Saeed, M. Political Optimizer: A novel socio-inspired meta-heuristic for global optimization. *Knowl.-Based Syst.* **2020**, *195*, 105709. [[CrossRef](#)]
37. Moraliyage, H.; Mills, N.; Rathnayake, P.; De Silva, D.; Jennings, A. UNICON: An Open Dataset of Electricity, Gas and Water Consumption in a Large Multi-Campus University Setting. In Proceedings of the 2022 15th International Conference on Human System Interaction (HSI), Melbourne, Australia, 28–31 July 2022; pp. 1–8. [[CrossRef](#)]
38. NVIDIA. cuBLAS. 2023. Available online: <https://docs.nvidia.com/cuda/cublas/> (accessed on 1 November 2024).

39. Appleyard, J. Optimizing Recurrent Neural Networks in cuDNN 5. 2026. Available online: <https://developer.nvidia.com/blog/optimizing-recurrent-neural-networks-cudnn-5/> (accessed on 1 November 2024).
40. Criado-Ramón, D.; Cojocaru, M.R.; Ruiz, L.G.B.; Servadei, L.; Wille, R.; Cuéllar, M.P.; Pegalajar, M.C. CUDA MHNN Energy Forecasting: CUDA Implementation of Metaheuristic and Memetic Neural-Network Training for Energy Forecasting. 2026. Available online: <https://github.com/xKuZz/cuda-mhnn-energy-forecasting> (accessed on 9 July 2026) .

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.