Artificial Neural Networks for Resources Optimization in Energetic Environment

Gianni D'Angelo (✉ giadangelo@unisa.it)

University of Salerno: Università degli Studi di Salerno

Francesco Palmieri

University of Salerno: Università degli Studi di Salerno

Antonio Robustelli

University of Salerno: Università degli Studi di Salerno

Research Article

Keywords: Artificial Neural Network, Resources Planning Optimization, Energetic Environment, Energetic Generators, Microgrid System, Artificial Intelligence

Posted Date: June 1st, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-405315/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Version of Record: A version of this preprint was published at Soft Computing on January 21st, 2022. See the published version at <https://doi.org/10.1007/s00500-022-06757-x>.

Artificial Neural Networks for Resources Optimization in Energetic Environment

Gianni D'Angelo* · Francesco Palmieri · Antonio Robustelli

Received: date / Accepted: date

Abstract Resource Planning Optimization (RPO) is a common task that many companies need to face to obtain several benefits, like budget improvements and run-time analyses. It is often addressed by using several software products and tools, based on sophisticated mathematical artifacts. However, these tools are not able to provide a practical solution because they are often expensive and time-consuming. On the other hand, Artificial Intelligence-based approaches have been increasingly used in many industrial and scientific fields in last decades, and have demonstrated to be a valid alternative to the classical mathematical-based methods. For this purpose, the following paper aims to investigate the use of multiple Artificial Neural Networks (ANNs) for solving a RPO problem related to the scheduling of different Combined Heat & Power (CHP) generators. The experimental results, carried out by using data extracted by considering a real Microgrid system, have confirmed the effectiveness of the proposed approach. Additionally, we show that multiple neural networks achieve up to a 6% improvement in average accuracy over Naive Bayes classifier, up to a 12% over Multi-Layer Perceptron classifier and up to a 13% over state-of-the-art ANNs in the presence of unbalanced training dataset.

* Corresponding author

G. D'Angelo
Department of Computer Science, University of Salerno, Italy
E-mail: giadangelo@unisa.it

F. Palmieri
Department of Computer Science, University of Salerno, Italy
E-mail: fpalmieri@unisa.it

A. Robustelli
Department of Computer Science, University of Salerno, Italy
E-mail: arobustelli@unisa.it

Keywords Artificial Neural Network · Resources Planning Optimization · Energetic Environment · Energetic Generators · Microgrid System · Artificial Intelligence

1 Introduction

Currently, Resource Planning Optimization (RPO) is a frequent task that many companies try to face to obtain many benefits [1], like budget improvements, run-time analyses, and human resource organization. Additionally, RPO becomes crucial in industrial contexts, because it can guarantee the operation of many industrial systems and machines, like actuators, generators, and sensors. Microgrids systems are a particular instance of them, where the resources to optimize are used 24 hours a day. More precisely, a Microgrid system is defined as a group of distributed energy resources that acts as a single controllable entity to provide energy for a local community, by satisfying three goals: reliability (physical and technological), sustainability (environmental considerations), and economics (cost optimizing and efficiency) [2, 3]. Additionally, in order to achieve these goals, Microgrid systems have got two modes called *stand-alone mode* and *grid-connected mode*, able to manage the energy by taking several high-level decisions [4–6]. For this purpose, several RPO models have been proposed for them [7–9]. However, RPO problems are often addressed by using several expensive and time-consuming software products and tools, such as Mavenlink [10], Enterprise Resource Management (ERM) [11], and Tempo Planner [12], which are based on sophisticated mathematical artifacts, and consequently, rarely provide a practical solution.

On the other hand, Artificial Intelligence (AI) has been increasingly used in many industrial and scientific fields in last decades, demonstrating to provide valid alternatives to the classical optimization approaches based on mathematical solutions. Indeed, it has been successfully used in many complex issues such as: cancer diagnosis [13], aerospace's structures testing [14], and malware detection [15]. The most precious ability of AI is learning complex patterns, features and relationships from huge amounts of data, independently from a specific domain, through several kinds of inference, classification and knowledge elicitation mechanisms falling under the umbrella of Machine Learning (ML) techniques. In particular, Artificial Neural Networks (ANNs) are able to solve many multi-classes problems as one, which is also called *multi-labels multi-classes problem*, and the usage of multiple ANNs has been employed in several research areas like robotics [16] and computer vision [17]. Also, since ANNs are able to provide a prediction rapidly, they could be used to achieve a practical solution, respect to used tools, and address several RPO problems.

Hence, the goal of the following proposal is to investigate the use of multiple ANNs for solving a RPO problem related to the scheduling of different Combined Heat & Power (CHP) generators. More precisely, a RPO related to a Microgrid system is presented. Then, experimental results, carried out

by using data obtained from the considered real scenario, are discussed and compared with the most famous ML-based approaches.

The remaining of the paper is organized as follows. Section 2 will show an overview of related works. Section 3 will present an overview of key concepts about Machine Learning and its related techniques. Section 4 will present a description of the optimization problem to be solved, that is formalized through a mixed-integer linear programming model. Section 5 will report the experimental setting, results, and comparison with state-of-art models. Finally, Section 6 will show the conclusions and future works.

2 Related Works

Since RPO problems are often faced in industrial contexts, Microgrid systems are a particular instance of them, where the resources to optimize are used 24 hours a day to ensure the correct operation of different mechanical artifacts typologies. For this purpose, many RPO models for these systems by considering their goals and their characteristics, have been proposed [7–9]. In 2015 A. Khodaei et al. proposed a microgrid planning problem under uncertain physical and financial information where it is decomposed into an investment master problem and an operational sub-problem. Here, the optimal planning decisions determined in the master problem are employed in the sub-problem to examine the optimality of the master solution by calculating the worst case optimal operations [18]. In 2016 W. Yuan et al. proposed a planning problem to coordinate the resource allocation with the objective of minimizing the system damage in case of natural disasters. In particular, a robust optimization based framework that considers uncertain natural disaster occurrence is proposed to coordinate the planning of distribution systems, and a computational algorithm is developed for solving this model [19]. In 2017 A. Khodaei proposed a new class of microgrids called provisional microgrids, based on provisional information. In particular, the provisional microgrid planning problem is defined, formulated and solved by considering the interactions among the provisional microgrid, the coupled microgrid, and the utility grid [20]. Unfortunately, the software that are often used by the companies, are based on these models and are usually expensive and do not always represent a useful solution in practice.

In addition, several Swarm Intelligence (SI) approaches have been proposed in many research fields. In fact, These systems consist of a population of simple agents that interact locally with one another and with their environment. The inspiration often comes from nature, and examples of swarm intelligence in natural systems include ant colonies, bird flocking, hawks hunting, animal herding, bacterial growth and so on. In 2015 L. Zuo et al. proposed an optimization scheduling method based on the Ant Colony algorithm in Cloud Computing environment. Here, a multi-objective optimization scheduling method is proposed in accordance with the make span and the user's budget costs as constraints, and it is solved with an improved version of Ant Colony algorithm

[21]. In 2019 L. LI et al. proposed a new particle swarm optimization algorithm to obtain an adaptive resource scheduling for multi-objective problem. In particular, it is translated into a set of sub-problems by using a proposed hybrid decomposition approach [22]. In 2020 L. Zhang et al. proposed an adaptive strategy for the Microgrids to optimize the droop control based on particle swarm optimization. In particular, a new fuzzy inference system is discussed to adjust the learning factor and inertia weight of the employed algorithm [23]. However, Swarm intelligence methods are based on complex mathematics structures and cannot therefore be exploited.

In the last decade, Artificial Intelligence-based approaches have been increasingly used in many industrial and scientific fields, and have demonstrated to be a valid alternative to the classical approach based on mathematical solutions. In fact, we can find two important contributions in 2017. Firstly, A. Tesfaye et al. proposed a new wind power forecasting method based on the combination of measured data from SCADA and an Artificial Neural Network (ANN) model. The achieved results by the employed fully-connected neural network have shown an average Accuracy of 86% [24]. Secondly, G. Villarrubia et al. proposed the use of ANNs to approximate the objective function in the optimization problems by using a non-linear regression. More precisely, the authors have proposed several experiments to minimize or maximize different basic objective functions. The achieved results have shown an average Accuracy of 97% [25].

3 Machine Learning and related techniques

One of the Artificial Intelligence (AI) abilities is to learn complex patterns, features, and relationships from numerous amounts of data also called Big Data. More precisely, Machine Learning, Deep Learning, and Data Mining-based approaches represent the basis for the application of AI in knowledge discovery processes. Generally, an AI-based approach tries to learn information by imitating the actions of an expert, like a child who imitates the actions of an adult [15]. The comparison between the goal to be achieved and the outcome derived by the machine state represents the core of the learning process of the intelligent machine. This approach has proven to be effective in many research areas [26–28]. However, when we try to build ML models, we can find more issues, especially during the training phase. It is often possible to work with ML models which present overfitting/underfitting training behaviors, unbalanced dataset, and high training time. For this purpose, we will discuss optimization techniques that are able to reduce overfitting and improve training time, regardless of the type of dataset employed (balanced/unbalanced). Subsequently, we will discuss evaluating algorithms often used to validate AI methods many times by considering several dataset partitions.

3.1 Regularization techniques

Since deep ANNs are highly flexible models, overfitting is an issue than can often arise when training them. For this reason, Regularization techniques are an important group of methods specifically aimed at reducing overfitting in ML models. These methods try to modify the learning algorithm to reduce the model's variance, and consequently, obtain a model able to extract as many relevant features as possible [29].

The most common type of regularization for deep learning models is the one that keeps the weights of the network small. This regularization technique is called weight regularization and has two different variations: *L1 regularization* and *L2 regularization* [29]. Instead, *Dropout* is another famous regularization technique that works by randomly removing nodes during the training phase. More precisely, Dropout sets up a probability value for each node to determine its chance to be included in the training at each iteration of the learning algorithm. It means that some nodes do not take part in the parameter-updating process, and consequently, the Backpropagation (BT) algorithm can compute the derivatives on a smaller network, with some nodes removed. However, it is important to keep in mind that, in order to evaluate the performance of the model on the test set, the original network is always used [29].

In this study, we use several Dropout layers to reduce overfitting for each employed ANNs in order to obtain the best possible results.

3.2 Resampling techniques

The Real-world scenarios are usually described by unbalanced data that do not represent classes equally. In most of the cases, they are another problem that can affect the training process of deep models. For this purpose, when an employed dataset consists of unbalanced data, more guidelines suggest to try different solutions, like *resampling techniques* [30], and *weighted classification* [31].

Resampling techniques are a set of methods that allow the transformation of an unbalanced dataset in a balanced dataset by adding or removing several instances. However, they try to adapt the dataset to the ML model, and consequently, the model generalization can be often limited. [32,33,30]. Vice-versa, weighted classification tries to adapt ML models by considering the frequency of each class. In this way, it is possible to set a frequency-weight value for each class without modifying the real dataset structure and limiting the model generalization [34,31].

Since we consider a real scenario related to Microgrid system, we use weighted classification to adapt each proposed ANN and improve the achieved results.

3.3 Evaluating techniques

One of the best practices to evaluate ML and DL models is to consider several data partitions. For this purpose, there are two methods able to get a very accurate estimation of a model's performance: *K-Fold cross-validation* and *Leave-One-Out cross-validation*. The key concept of the following algorithms consists of to evaluate the developed model by considering several partitions of the employed dataset, instead of dividing it into two mutually-exclusive subsets.

In K-Fold cross-validation, the dataset is randomly split into k approximately equal-size subsets or folds. In the beginning, the first fold is used as a test set, and the model is trained on the remaining $k - 1$ folds. This process is repeated k times, and a different fold is used as the test set in each iteration, while the remaining $k - 1$ folds are employed as the training set. In practice, K-Fold cross-validation algorithm is usually performed with $k = 5$ or $k = 10$, because they are the recommended value to obtain a good validation of developed models [29].

Instead, in Leave-One-Out (LOO) cross-validation only one single piece of data example is used for evaluation. If there are n data examples the model is trained on $n - 1$ samples, while the single remaining example is used to compute the test error rate [29].

Since LOO cross-validation can be seen as a particular case of K-Fold cross-validation if $k = n$, we will use K-Fold algorithm to validate the proposed ANNs on several partitions of the employed dataset.

4 Microgrid Optimization Problem

Since Microgrid systems provide energy for a local community by satisfying several goals like cost optimization and efficiency [2,3], we chose to address a RPO problem related to the optimal scheduling of different CHP generators in a Microgrid system. More precisely, it can be stated as a mixed-integer linear programming model including a cost function J to minimize, and several constraints to fulfil, such as storage model constraints and operating conditions. Consequently, the main goal of the following RPO problem consists of to provide a resource plan for employed generators by considering several aspects, like input parameters and constraints. Predictions of the upcoming demand, the energy available, energy prices, and the production from renewable energy units, define input parameters. Instead, constraints determine the quantitative of imported/exported energy from/to the grid, when buy/sell the energy, and how/when to use generators. For this purpose, there are many types of constraints in the considered RPO problem, like interactive, operative and physical constraints.

Therefore, in order to provide a high-level definition of the following problem, we report the cost function J definition, interaction constraints respect to the main grid, and operating conditions.

4.1 Cost Function

The resource optimization, referred to Microgrid systems, is achieved by designing the decision variables so that a cost function, representing the operating costs, is minimized. For this purpose, the definitions of logical and decision variables are reported in order to introduce the cost function.

Let k a time instance, T the length of the prediction horizon, and N_g the number of generators. The state δ of a Distributed Generator (DG) and the power level P of a DG, the start-up costs SU of a DG, and the shut-down costs SD of a DG, are defined, for the i -th generator and for each time instance k , as follows:

$$\delta_i(k) = \begin{cases} 1 & \text{if the } i\text{-th DG is on} \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned} P_i(k) &\geq 0, \\ SU_i(k) &\geq 0, \\ SD_i(k) &\geq 0, \end{aligned}$$

with $i = 1, \dots, N_g$ and $k = 0, \dots, T - 1$.

Therefore, the minimum cost function J including costs associated with energy production, start-up and shut-down decisions, and possible earnings and curtailment penalties, is defined as follows [35]:

$$\begin{aligned} J := & \sum_{k=0}^{T-1} \sum_{i=1}^{N_g} [C_i^{DG}(P_i(k)) + OM_i \delta_i(k) + SU_i(k) + SD_i(k)] \\ & + C^{grid}(k) + \rho_c \sum_{h=1}^{N_c} \beta_h(k) D_h^c(k) \end{aligned}$$

where J is a quadratic cost function and the descriptions of each parameter are shown in Table 1.

Therefore, according to the function J , the following decisions could be taken by a Microgrid system:

- when each generation unit should be started and stopped, and how much each unit should generate to meet this load at minimum cost. The cost of these choices is considered, at each instant i , by the following sum: $\sum_{i=1}^{N_g} [C_i^{DG}(P_i(k)) + OM_i \delta_i(k) + SU_i(k) + SD_i(k)]$.

- when and how much energy should be purchased or sold to the main grid. This cost is represented by $C^{grid}(k)$.
- when and which controllable loads must be shed/curtailed. The cost of these choices is considered, at each instant h , by the following sum: $\rho_c \sum_{h=1}^{N_c} \beta_h(k) D_h^c(k)$.

Parameter	Description
k	Current time instant
T	Length of the prediction horizon
DG	Distributed Generator
N_g	Number of DG units
N_c	Number of controllable loads
P	Power level of a DG unit [kW]
$C^{DG}(P)$	Fuel consumption cost curve of a DG unit
OM	Operating cost of a DG unit [€/kWh]
δ	State of a DG unit [off(0)/on(1)]
$SU_i(k)$	Start-up cost for the i^{th} DG unit
$SD_i(k)$	Shut-down cost for the i^{th} DG unit
$C^{grid}(k)$	Earning/Penalty for the microgrid system
ρ_c	Penalty weight on curtailments
β	Curtailed power percentage
D^c	Preferred power level of a controllable load [kW]
P^g	Importing/exporting power level [kW]
δ^g	Importing/exporting mode from/to the main grid
c^P	Purchasing energy prices [€/kWh]
c^S	Selling energy prices [€/kWh]
T^{up}	Minimum up time of a DG unit [h]
T^{down}	Minimum down time of a DG unit [h]
c^{SU}	Start-up cost of a DG unit [€]
c^{DU}	Shut-down cost of a DG unit [€]

Table 1: Parameters

4.2 Interaction Constraints

Generally, each Microgrid system has got a special mode called *grid-connected mode*, which is able to purchase and sell energy continuously and take many high-level decisions [4–6]. When the grid-connected mode is on a Microgrid system is able to purchase energy from the main grid by respecting the interaction constraints. These constraints play a fundamental role in a Microgrid system because they define the rules applied to purchase/sell the energy from/to the main grid. For this purpose, the interaction constraints are defined, at each instant k , by considering the importing/exporting power level $P^g(k)$ from the main grid [35]. More precisely, if $P^g(k)$ is greater than zero the energy is purchased $\delta^g(k) = 1$, otherwise the energy is sold $\delta^g(k) = 0$.

Let k a time instance and T the length of the prediction horizon. The importing/exporting power level P^g from/to the main grid, and the import-

ing/exporting mode δ^g from/to the main grid, are defined, for each time instance k , as follows:

$$P^g(k) = \begin{cases} > 0 & \text{if the power is imported} \\ < 0 & \text{otherwise} \end{cases}$$

$$\delta^g(k) = \begin{cases} 1 & \text{if the importing mode is on} \\ 0 & \text{otherwise} \end{cases}$$

with $k = 0, \dots, T - 1$.

Then, the following interaction constraints to purchase/selling energy from/to the main grid are defined, for each time instance k , as follows:

$$P^g(k) > 0 \iff \delta^g(k) = 1$$

Consequently, let c^P and c^S the purchasing and selling energy prices, respectively. The cost of the imported/exported energy C^g is defined, for each time instance k , as follows:

$$C^g(k) = \begin{cases} c^P(k)P^g(k) & \text{if } \delta^g(k) = 1 \\ c^S(k)P^g(k) & \text{otherwise} \end{cases}$$

4.3 Operating Conditions

On the other hand, since a Microgrid system consists of several generators, there is the necessity to establish rules about how and when each generator should be used. These rules are defined by operating constraints that are able to consider, at any time, the state of each generator, and their start-up and shut-down costs. For this purpose, they can be defined, at each instant k , on the minimum amount of time for which a generator must be kept on/off (minimum up/down times) [35]:

$$\delta_i(k) \geq \delta_i(k - t_{up} - 1) - \delta_i(k - t_{up} - 2),$$

$$1 - \delta_i(k) \geq \delta_i(k - t_{down} - 2) - \delta_i(k - t_{down} - 1),$$

with $i = 1, \dots, N_g$, $t_{up} = 0, \dots, \min(T_i^{up}, k - T_i^{up} + 2)$, and $t_{down} = 0, \dots, \min(T_i^{up}, k - T_i^{up} + 2)$.

Consequently, the start-up and shut-down cost for the i^{th} generator can be expressed, at each instant k , as follows:

$$\begin{aligned}
SU_i(k) &\geq c_i^{SU}(k)[\delta_i(k) - \delta_i(k-1)], \\
SD_i(k) &\geq c_i^{SD}(k)[\delta_i(k-1) - \delta_i(k)], \\
SU_i(k) &\geq 0, \\
SD_i(k) &\geq 0,
\end{aligned}$$

with $i = 1, \dots, N_g$.

Finally, since the following discussion has been done to provide a high-level definition of the proposed problem, we remand to [35] for more details about the already discussed operating conditions, the definitions of physical constraints, and other theoretical assumptions about the considered problem.

4.4 Problem Instance

In the considered scenario, the resources to be scheduled are five special generators named Jenbacher, Caterpillar, and three Chiller, respectively [36,37]. As reported in Tables 2, 3, 4, and 5, each of them has got four different working modes with an associated cost, and additionally, they are able to satisfied two energy typologies requests: electrical and thermal. In this instance, Jenbacher and Caterpillar generators are able to produce both electricity and thermal energy, whereas the remaining three Chiller generators can only produce thermal energy. Also, the request to satisfy, at each hour, is formulated on the basis of the following three parameters: the required electricity quantitative, the required thermal quantitative, and the available quantitative of electricity. More precisely, the required electricity and thermal quantitative represent the prediction of upcoming demand, that is satisfied according to the presented interaction constraints and operating conditions. Instead, the available quantitative of electricity is a free quantitative of energy obtained from renewable energy sources, and that it is often used to minimize the required costs.

Therefore, the goal of this RPO problem is to compute the optimal scheduling of the described generators, satisfying the energetic request at each hour, and consequently, by minimizing the cost function J .

Parameter	Values
Number of modes	4
Max electric power (kWh)	1000
Max thermal power (kWh)	672.56
Electric modes (kWh)	1000 642.32 424.77 427.90
Electric costs (€/kWh)	0.096 0.096 0.101 0.100
Thermal modes (kWh)	672.56 605.30 538.05 470.70
Thermal costs (€/kWh)	0.048 0.046 0.065 0.061

Table 2: Parameters of the Jenbacher generator

Parameter	Values
Number of modes	4
Max electric power (kWh)	600
Max thermal power (kWh)	362.09
Electric modes (kWh)	600 540 480 420
Electric costs (€/kWh)	0.081 0.086 0.089 0.092
Thermal modes (kWh)	362.09 374.82 343.68 311.10
Thermal costs (€/kWh)	0.053 0.051 0.052 0.054

Table 3: Parameters of the Caterpillar generator

Parameter	Values
Number of modes	4
Max thermal power (kWh)	805.80
Thermal modes (kWh)	805.80 604.35 402.90 201.45
Thermal costs (€/kWh)	0.051 0.043 0.037 0.034

Table 4: Parameters of the Chiller 1 and 2 generators

Parameter	Values
Number of modes	4
Max thermal power (kWh)	678.90
Thermal modes (kWh)	678.90 509.17 339.40 169.72
Thermal costs (€/kWh)	0.200 0.052 0.053 0.053

Table 5: Parameters of the Chiller 3 generator

5 Experimental results

The goal of the experiment reported, in this section, is devoted to demonstrating the possibility of multiple ANNs to compute a resource scheduling related to the considered Microgrid system. For this purpose, the presented experimental results have been carried out by using data extracted by a real scenario focused on an energetic environment.

5.1 Dataset and Experimental Setting

The dataset used during the experimental evaluation consists of historical data which have been collected, for two years, from a Matlab solver develop by Ital-data [38]. In particular, more than 10000 rows have been stored, and each of them presents 12 different values reported by Tables 2, 3, 4, and 5, respectively. The first five values represent the input data, and each value means the available quantitative of electricity, the required electricity quantitative, the required thermal quantitative, the hour of the day, and the day, respectively. Instead, the last seven represent the output data, and each value means the power modes for each generator.

Therefore, since the employed dataset consists of many data and several input/output values typologies, an Exploratory Data Analysis (EDA) technique has been used in order to obtain a complete overview of them [39, 40].

This approach is able to analyze a dataset, and consequently, summarize its main characteristics by using several charts, as they are shown in Figures 1, 2, 3, and 4.

Fig. 1: Boxplot representation of input data distribution.

Fig. 2: Distribution of electricity output categories of Jenbacher and Caterpillar generators.

In Figure 1 is show the boxplot representation of input data distribution. In particular, five boxplots are reported for each input data, respectively. The first one shows the data distribution of available energy. The second and third one report the distribution of the required electricity and thermal quantita-

Fig. 3: Distribution of thermal output categories of Jenbacher and Caterpillar generators.

Fig. 4: Distribution of thermal output categories of Chiller generators.

tive, respectively. Finally, the last two show the hours and days when each demand has been satisfied. Instead, Figures 2, 3, and 4 shows the distributions of output categories for each generator. More precisely, in the first one is shown the distribution of electricity output categories of Jenbacher and Caterpillar generators, in the second one is shown the distribution of thermal output categories of Jenbacher and Caterpillar generators, while in the third one is shown the distribution of thermal output categories of Chiller generators. Additionally, due following charts, it is possible to highlight the nature extremely unbalanced of the employed dataset.

Subsequently, we have split the following dataset in order to run the experiments. For this purpose, the whole dataset has been subdivided into two mutually exclusive subsets called training and testing dataset, respectively. We used 70% of the entire dataset for training and the remaining 30% for testing. Then, the K-Fold cross-validation algorithm, with $k=5$, has been used to tune the hyper-parameters and provide an unbiased evaluation of each employed

ANN. More precisely, we used $k=5$ because it is a recommended value [29], and consequently, the entire dataset has been, in turn, equally partitioned in five training and testing sets. Finally, each ANN has been trained on each training set and evaluated on the corresponding testing set.

5.2 Proposed Networks and Evaluation Metrics

Since Jenbacher and Caterpillar generators are able to produce both electricity and thermal energy, they have got two working modes to satisfy the required electricity and thermal quantitative, respectively. For this purpose, seven ANNs, one for each work mode presented earlier in Subsection 4.4, have been developed as a fully-connected neural network composed of two Dense layers with 800 neurons, activation=relu, and Dropout=0.5. Additionally, each network had a Dense output layer with activation=softmax and a number of nodes equal to the number of output categories expected. The proposed architecture has been derived from training and testing processes in order to obtain the best results as possible. More precisely, we tested different hyper-parameters like the number of layers in a network (2,3,4,5), the number of neurons for each layer (100,200,400,800), the typologies of activation functions (relu, tanh, sigmoid, and softmax), different values of batch_size (32,64,128,256), the optimized used to train each model (Stochastic Gradient Descent - SGD, Adam, and Adamax), and the loss functions (Mean Absolute Error - MAE, Mean Squared Error - MSE, and categorical_crossentropy). Figure 5 shows the high-level architecture of the proposed networks, while Table 6 summarizes their main information.

Fig. 5: High-level architecture of each network.

Network	Generator	Typology	Alias	Categories
Net 1	Jenbacher	Electric	JE	5
Net 2	Caterpillar	Electric	CE	5
Net 3	Jenbacher	Thermal	JT	5
Net 4	Caterpillar	Thermal	CT	5
Net 5	Chiller 3	Thermal	C3	4
Net 6	Chiller 1	Thermal	C1	4
Net 7	Chiller 2	Thermal	C2	4

Table 6: Description of each network

To evaluate the classification quality of the employed networks, the following metrics have been computed: Accuracy (Acc.), Sensitivity (Sens.), Specificity (Spec.), Precision (Prec.), Area Under the ROC Curve (AUC), and F-Measure (F-Meas or F-Score). More precisely, they have been derived from a multi-class confusion matrix where, for each category, TPs (True Positives) are the instances correctly classified, TNs (True Negatives) are the instances correctly classified in another category, FPs (False Positives) are the instances incorrectly identified as a considered category, while FNs (False Negatives) are the instances in another category incorrectly identified as a considered category. Subsequently, to obtain a global validation, the average values (Avg.) and standard deviation values (Dev.), among all metrics, have been computed.

5.3 Archived Results and Discussion

The proposed ANNs have been trained and tested with a PC-Laptop equipped with an Intel 4-Core I5-8265U CPU @ 1.60GHz, and 8 GB RAM. Each employed neural network has been compiled with Adam optimizer and categorical_crossentropy loss function. Then, they have been trained with *batch_size* = 256, weighted classification technique, and 1000 epochs by using the 70/30 criteria and the K-Fold cross-validation algorithm with k=5. The following hyper-parameters have been chosen according to the achieved results from the testing process. Table 7 shows the evaluation metrics for each ANN. Table 8 shows the average (Avg.) and standard deviation values (Dev.) of each evaluation metric, respectively. Instead, Table 9 shows the validation results obtained by performing K-Fold cross-validation algorithm with k=5.

	Acc.	Spec.	Prec.	Sens.	F-Score	AUC
Net 1	1.0000	1.0000	1.0000	1.0000	1.0000	0.9992
Net 2	0.9995	0.9998	0.9891	0.9958	0.9924	0.9988
Net 3	1.0000	1.0000	1.0000	1.0000	1.0000	0.9993
Net 4	0.9993	0.9996	0.9950	0.9941	0.9945	0.9989
Net 5	1.0000	1.0000	1.0000	1.0000	1.0000	0.9994
Net 6	0.9999	1.0000	0.9981	0.9999	0.9990	0.9993
Net 7	0.9999	1.0000	0.9978	0.9999	0.9988	0.9993

Table 7: Evaluation metrics for each network

	Acc.	Spec.	Prec.	Sens.	F-Score	AUC
Avg.	0.9998	0.9999	0.9971	0.9985	0.9978	0.9992
Dev.	0.0003	0.0002	0.0040	0.0025	0.0031	0.0002

Table 8: Average and deviation values for each metric

	Acc.	Spec.	Prec.	Sens.	F-Score	AUC
Avg.	0.9690	0.9784	0.9030	0.8809	0.8598	0.9905
Dev.	0.0291	0.0191	0.0652	0.0805	0.0900	0.0084

Table 9: K-Fold cross validation global average results

In order to show the effectiveness of the employed ANNs, a comparison, between most notable ML-based approaches, has been made by using WEKA [12]. More precisely, we used *Multi-Layer Perceptron* (MLP), *J48* trees (J48), and *Naive Bayes* (NB), to derive the classification metrics for each work mode identified by an Alias in Table 6. The obtained results are respectively shown in Tables 10, 11, and 12, while Table 13 summarizes the comparison between the proposed ANNs and the used ML-based methods.

Alias	Acc.	Spec.	Prec.	Sens.	F-Score	AUC
JE	0.8770	0.8770	0.5480	0.8770	0.5744	0.8770
CE	0.8010	0.8010	0.3356	0.8010	0.4729	0.8010
JT	0.8790	0.8790	0.5552	0.8790	0.6803	0.8790
CT	0.8960	0.8960	0.5012	0.8960	0.6427	0.8960
C1	0.9360	0.9360	0.4633	0.9360	0.6198	0.9360
C2	0.9150	0.9150	0.6918	0.9150	0.7879	0.9150
C3	0.9190	0.9190	0.6960	0.9190	0.7948	0.9150

Table 10: Metrics for MLP classifier

Alias	Acc.	Spec.	Prec.	Sens.	F-Score	AUC
JE	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
CE	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
JT	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
CT	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
C1	0.9891	0.9891	0.9891	0.9891	0.9891	0.9891
C2	0.9933	0.9933	0.9933	0.9933	0.9933	0.9933
C3	0.9927	0.9927	0.9927	0.9927	0.9927	0.9927

Table 11: Metrics for J48 classifier

The following comparison shows that the NB and MLP classifiers have achieved discrete results, while J48 has obtained excellent evaluation metrics. However, the employed ANNs have achieved the best results in solving of

Alias	Acc.	Spec.	Prec.	Sens.	F-Score	AUC
JE	0.9950	0.9950	0.9950	0.9950	0.9950	0.9950
CE	0.9660	0.9670	0.9660	0.9660	0.9660	0.9665
JT	0.9950	0.9950	0.9950	0.9950	0.9950	0.9950
CT	0.9660	0.9670	0.9660	0.9660	0.9660	0.9665
C1	0.8880	0.8880	0.8920	0.8880	0.8920	0.8880
C2	0.8980	0.8980	0.9090	0.8980	0.9000	0.8980
C3	0.8250	0.8250	0.8740	0.8250	0.8440	0.8250

Table 12: Metrics for Naive Bayes classifier

	Acc.	Spec.	Prec.	Sens.	F-Score	AUC
ANNs	0.9998	0.9999	0.9971	0.9985	0.9978	0.9992
J48	0.9963	0.9963	0.9963	0.9963	0.9963	0.9963
NB	0.9333	0.9336	0.9510	0.9333	0.9454	0.9333
MLP	0.8890	0.8890	0.5416	0.8890	0.8531	0.8890

Table 13: Comparison between most notable ML methods

considered RPO problem. More precisely, they have archived up to a 6% improvement in average accuracy over Naive Bayes classifier and up to a 12% over Multi-Layer Perceptron classifier. Additionally, their evaluation metrics (like Precision, Sensibility, and F-Score) are higher than those obtained from the J48 decision trees, as it happened in several studies [41–43]. More precisely, they have reported many comparisons between ANNs and decision trees where the employed neural networks have achieved better results than decision trees. Finally, the proposed approach has archived up to a 13% improvement in average accuracy over state-of-the-art ANN for power forecasting [24]. For this purpose, the proposed approach can reduce the number of FPs and FNs, and consequently, better minimize the classification error respect to the most famous ML-based approaches and the proposed state-of-the-art models. Also, it could be applied in several research fields to investigate many RPO problems.

6 Conclusions and future works

In this paper, multiple Artificial Neural Networks (ANNs) have been employed for solving a Resource Planning Optimization (RPO) problem related to the the scheduling of different Combined Heat & Power (CHP) generators for a Microgrid system. For this purpose, we have defined it as a mixed-integer linear programming model characterized by a minimum cost function. Subsequently, seven different ANNs have been used and validated by using statistic metrics, and additionally, we have compared them to the most famous Machine Learning approaches provide by WEKA. The obtained results show that the proposed ANNs have archived up to a 6% improvement in average accuracy over Naive Bayes classifier, up to a 12% over Multi-Layer Perceptron classifier, and up to a 13% over state-of-the-art ANNs, and consequently, could be a practical approach to solve the considered problem.

For this purpose, we would like to propose two possible future works. First of all, in order to validate the employed approach, we will explore new RPO problems by considering several and critical scenarios. Second, since a specific ANN has been employed for each working mode, we will investigate new AI models to use a dedicated neural network for each generator. For instance, a Recurrent Neural Network (RNN) could be a good method for use temporal features, like the hour of the day. Moreover, the Autoencoders (AEs) could be investigated to extract relevant features are able to reduce the number of employed networks.

Compliance with Ethical Standards

Ethical approval. This article does not contain any studies with animals performed by any of the authors.

Funding. No funding was received for conducting this study.

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

Informed consent. Informed consent was obtained from all individual participants included in the study.

Authorship contributions

Gianni D'Angelo: Investigation, Conceptualization, Methodology, Supervision, Writing - review and editing.

Francesco Palmieri: Conceptualization, Supervision, Writing - review and editing-

Antonio Robustelli: Formal analysis, Investigation, Software, Writing - original draft, Preparation.

References

1. Halima. Five benefits of resource scheduling. <https://www.visual-planning.com/en/blog/five-benefits-resource-scheduling> (2017)
2. A. Hirsch, Y. Parag, J. Guerrero, Renewable and Sustainable Energy Reviews **90**, 402 (2018). DOI <https://doi.org/10.1016/j.rser.2018.03.040>. URL <http://www.sciencedirect.com/science/article/pii/S136403211830128X>
3. E. Hossain, E. Kabalcı, R. Bayindir, R. Perez, International Journal of Renewable Energy Research **4**, 1094 (2014)
4. Q. Jiang, M. Xue, G. Geng, IEEE Transactions on Power Systems **28**(3), 3380 (2013). DOI 10.1109/TPWRS.2013.2244104

5. D. Das, G. Gurralla, U.J. Shenoy, *Sādhanā* **42**(8), 1239 (2017). DOI 10.1007/s12046-017-0659-z. URL <https://doi.org/10.1007/s12046-017-0659-z>
6. V. Joseph, P.C. Thomas, in *2013 International Conference on Advanced Computing and Communication Systems* (2013), pp. 1–6. DOI 10.1109/ICACCS.2013.6938738
7. C. Yuan, M.S. Illindala, A.S. Khalsa, *IEEE Transactions on Sustainable Energy* **8**(4), 1351 (2017)
8. K.S.A. Sedzro, A.J. Lamadrid, L.F. Zuluaga, *IEEE Transactions on Power Systems* **33**(3), 2633 (2018)
9. S. Mohamed, M.F. Shaaban, M. Ismail, E. Serpedin, K.A. Qaraqe, *IEEE Transactions on Sustainable Energy* **10**(1), 257 (2019)
10. Mavenlink. Resource scheduling software | mavenlink. <https://www.mavenlink.com/> (2020)
11. S. Software. Resource scheduling - powerful and highly configurable software. <https://www.saviom.com/resource-management-software/resource-scheduling-software.php> (2020)
12. Tempo. Jira resource management software and planning tool | tempo planner. <https://www.tempo.io/tempo-planner-jira-resource-planning-and-management> (2020)
13. S. Elia, G. D'Angelo, F. Palmieri, R. Sorge, R. Massoud, C. Cortese, G. Hardavella, A. De Stefano, *Soft Computing* **24**(10), 7281 (2020). DOI 10.1007/s00500-019-04344-1. URL <https://doi.org/10.1007/s00500-019-04344-1>
14. G. D'Angelo, S. Rampone, in *2015 IEEE Metrology for Aerospace (MetroAeroSpace)* (2015), pp. 406–410
15. G. D'Angelo, M. Ficco, F. Palmieri, *Journal of Parallel and Distributed Computing* **137**, 26 (2020). DOI <https://doi.org/10.1016/j.jpdc.2019.11.001>. URL <http://www.sciencedirect.com/science/article/pii/S0743731519302436>
16. H. Li, J. Li, X. Han, in *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)* (2019), pp. 2571–2577
17. S.N. Kanuri, S.P. Navali, S.R. Ranganath, N.V. Pujari, in *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)* (2018), pp. 1837–1842
18. A. Khodaei, S. Bahramirad, M. Shahidehpour, *IEEE Transactions on Power Systems* **30**(5), 2417 (2015)
19. W. Yuan, J. Wang, F. Qiu, C. Chen, C. Kang, B. Zeng, *IEEE Transactions on Smart Grid* **7**(6), 2817 (2016)
20. A. Khodaei, *IEEE Transactions on Smart Grid* **8**(3), 1096 (2017)
21. L. Zuo, L. Shu, S. Dong, C. Zhu, T. Hara, *IEEE Access* **3**, 2687 (2015)
22. L. Li, S. Chen, Z. Gong, Q. Lin, Z. Ming, *IEEE Access* **7**, 177082 (2019)
23. L. Zhang, H. Zheng, Q. Hu, B. Su, L. Lyu, *IEEE Access* **8**, 3579 (2020)
24. A. Eseye, J. Zhang, D. Zheng, D. Shiferaw, *International Journal of Science and Engineering Applications* **5**, 144 (2016). DOI 10.7753/IJSEA0503.1005
25. G. Villarrubia, J. De Paz, P. Chamoso, F. De La Prieta, *Neurocomputing* **272** (2017). DOI 10.1016/j.neucom.2017.04.075
26. G. D'Angelo, F. Palmieri, *Future Generation Computer Systems* **102**, 633 (2020). DOI <https://doi.org/10.1016/j.future.2019.09.007>. URL <http://www.sciencedirect.com/science/article/pii/S0167739X19306193>
27. G. D'Angelo, F. Palmieri, S. Rampone, *Information Sciences* **486**, 31 (2019). DOI <https://doi.org/10.1016/j.ins.2019.02.015>. URL <http://www.sciencedirect.com/science/article/pii/S0020025519301240>
28. G. D'Angelo, M. Tipaldi, F. Palmieri, L. Glielmo, *Information Sciences* **504**, 501 (2019). DOI <https://doi.org/10.1016/j.ins.2019.07.067>. URL <http://www.sciencedirect.com/science/article/pii/S0020025519306796>
29. R. Bhagwat, M. Abdolahnejad, M. Moocarme, *Applied deep learning with Keras: solve complex real-life problems with the simplicity of Keras* (Packt Publishing, 2019)
30. R. Ghorbani, R. Ghousi, *IEEE Access* **8**, 67899 (2020). DOI 10.1109/ACCESS.2020.2986809
31. Z. Xu, C. Dan, J. Khim, P. Ravikumar. Class-weighted classification: Trade-offs and robust approaches (2020)

32. J. Brownlee. 8 tactics to combat imbalanced classes in your machine learning dataset. <https://machinelearningmastery.com/tactics-to-combat-imbalanced-classes-in-your-machine-learning-dataset/> (2017)
33. A. Vidhya. Imbalanced data : How to handle imbalanced classification problems. <https://www.analyticsvidhya.com/blog/2017/03/imbalanced-data-classification/> (2017)
34. M. Hashemi, H. Karimi, *Statistics, Optimization and Information Computing* **6**(4), 497 (2018)
35. A. Parisio, L. Glielmo, in *2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)* (2011), pp. 505–510
36. Innio. Jenbacher gas engines. <https://www.innio.com/en/products/jenbacher> (2020)
37. Caterpillar. Produzione di energia elettrica con generatore di corrente. <https://www.cat.com/> (2020)
38. I. S.p.A. Italdata | services and solution for smart communities. <https://www.italdata.it/> (2020)
39. J. Weng. Exploratory data analysis: A practical guide and template for structured data. <https://towardsdatascience.com/exploratory-data-analysis-eda-a-practical-guide-and-template-for-structured-data-abfbf3ee3bd9> (2020)
40. T.N. Prabhu. Exploratory data analysis in python. <https://towardsdatascience.com/exploratory-data-analysis-in-python-c9a77dfa39ce> (2020)
41. S. Tharaha, K. Rashika, in *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIECS)* (2017), pp. 1–5. DOI 10.1109/ICIECS.2017.8276107
42. O. Karakurt, H. Erdal, E. Namlı, H. Yumurtaci-Aydogmus, Y. Turkkan, *Science Park* **1**, 1 (2013). DOI 10.9780/23218045/1172013/41
43. M.W. Ahmad, M. Mourshed, Y. Rezgui, *Energy and Buildings* **147**, 77 (2017). DOI <https://doi.org/10.1016/j.enbuild.2017.04.038>. URL <https://www.sciencedirect.com/science/article/pii/S0378778816313937>

Figures

Figure 1

Boxplot representation of input data distribution.

Figure 2

Distribution of electricity output categories of Jenbacher and Caterpillar generators.

Figure 3

Distribution of thermal output categories of Jenbacher and Caterpillar generators.

Figure 4

Distribution of thermal output categories of Chiller generators.

Figure 5

High-level architecture of each network.