ORIGINAL ARTICLE OPEN ACCESS

Artificial Intelligent Application in Project Management: An Algorithm Comparison for Solar Plants Planning Construction

Manuel Ángel López Ferreiro[1] | Jesús Gil Ruiz[2] | Óscar García[1] | Luis De La Fuente Valentín[1]

[1]UNIR, International University of La Rioja, Logroño, La Rioja, Spain | [2]Universidad Europea de Madrid, Madrid, Spain

Correspondence: Óscar García (oscar.garcia.garcia@unir.net)

Received: 5 March 2025 | **Revised:** 7 July 2025 | **Accepted:** 12 July 2025

Keywords: adaptive neuro-fuzzy inference | artificial intelligence | construction schedule | neuroevolution | optimization | project planning

ABSTRACT

Construction planning is a critical and complex phase in the deployment of large-scale renewable energy infrastructure. This study applies artificial intelligence techniques to a domain-specific problem that has traditionally relied on expert judgement: the generation of detailed construction schedules for photovoltaic power plants. As renewable generation is a key part to meet the challenges of energy transition, the implementation of large projects has increased in recent years and this trend is expected to continue in the future. The main difficulty in meeting construction deadlines is the elaboration of an adequate planning. A tool that automatically generates schedules can be of great help to set up an initial baseline planning. To this end, this work compares five artificial intelligence techniques, on a data set consisting of real examples of successfully completed projects. The evaluation of the results obtained on test data shows that Adaptive Neuro-Fuzzy Inference System (ANFIS) is the technique that obtains the best performance in all error metrics, although it entails a high computational cost. The model thus obtained manages to generate a complete construction schedule with an error of 8% of the total duration. The use of metrics as MAE, MSE and R^2 provides a robust understanding of prediction accuracy, variability, and fit. These metrics are commonly used in project planning evaluations and help interpret model behaviour under different error profiles. Additionally, the resulting 8% total duration error implies a deviation of around 24 days in a 300-day project, which is highly actionable in real-world solar project management. The findings not only demonstrate the feasibility of using AI for solar construction planning, but also lay the groundwork for the development of intelligent software tools or platforms that could support planners in the renewable energy sector. While this study focuses on photovoltaic plants, the approach is extendable to other power plants as wind farms, combined-cycle or nuclear plants, or even to other construction projects.

1 | Introduction

1.1 | Description of the Problem

Renewable generation is a key factor in meeting the challenges posed by the energy transition and satisfying growing energy demand in a sustainable manner. Renewable energy sources currently account for around 40% of the world's installed capacity and generate 28% of total electricity consumption. This has been made possible by the implementation of a large number of new projects, which has resulted in a sustained annual growth in installed renewable power of 9% over the last 10 years (IRENA 2023).

However, the targets for the use of energy from renewable sources require an even greater effort for the deployment of these infrastructures. In the European case, the EU Directive 2018/2001 foresees that the share of energy from renewable sources should be at least 32% of the EU's gross final energy consumption by 2030. This regulation must be transposed according to the characteristics of each country, although in order to understand with an example what this means in terms of the need to build new facilities, the case of Spain can be taken as a reference. There, the renewable production of the electricity system in 2021 was 45.8%, compared to an EU average of 36.9% (IRENA 2023). The Spanish National Integrated Energy and Climate Plan approved for 2021–2030 envisages increasing electricity generation from renewable sources to 74% by 2030. This plan, in its draft update for 2023, proposes to increase this coverage to 81%, due to the increase in climate ambition at the European level, embodied in the 'Objective 55' and 'REPowerEU' packages.

Given the demand for a rapid development of renewable capacity, the need for efficient planning in the construction of renewable plants emerges as a crucial component for the success of this transition. In practice, the construction of large renewable infrastructures is a major challenge in terms of organisation and management, although not, in general, in terms of technological complexity. As a result, the profit margins for this type of project are very narrow. This implies reduced construction time requirements, which translates into greater complexity due to the need to coordinate a large number of resources simultaneously, and also the need for strict control of investments and execution deadlines in order to ensure the profitability of these projects. It is therefore particularly important to develop optimal and realistic schedules that avoid biases and facilitate decision-making, since the most important root cause of delays in construction projects is inadequate planning (Agyekum-Mensah and Knight 2017).

Delays in the construction of utility-scale photovoltaic plants can have severe economic consequences. From the contractor's perspective, Engineering, Procurement, and Construction (EPC) agreements typically impose financial penalties for missed deadlines, which can amount to up to 20% of the total project budget. For a 50 MW photovoltaic plant, with an estimated cost of 35–45 million euros based on IRENA benchmarks (i.e., 700–900 euros/kW), this could translate into penalties of 7–9 million euros. In addition, construction delays increase labor costs, as teams and equipment must remain on-site beyond the planned timeline. On the revenue side, delays also mean lost income due to postponed energy sales under long-term power purchase agreements (PPAs). A typical 50 MW plant producing 85 GWh/year, under a PPA of 50 euros/MWh over 10 years, could miss out on 350,000–400,000 euros per month of delay. From an environmental standpoint, this also represents delayed reductions in CO_2 emissions. Assuming a grid emissions factor of 0.3–0.4 tCO_2/MWh, each month of delay would prevent the avoidance of 2100–2800 tons of CO_2. These factors reinforce the importance of accurate schedule planning and highlight the need for predictive, data-driven tools to minimise risk and uncertainty in renewable energy deployment.

Currently, planning of construction projects is done on the basis of expert judgement. Planning based on computational models is still in an experimental phase, without satisfactory results to date, due to the complexity of the domain and the difficulty of obtaining the information to feed these models (Regona et al. 2022). Thus, there are no relevant studies where construction planning is modelled from real data sets, nor even less a comparison between the performances of different models.

Recent advances in artificial intelligence (AI) have shown great potential in addressing complex decision-making problems in construction and engineering. For instance, Gong et al. (2024) developed a structured framework integrating customer and low-carbon demands to optimise remanufacturing schemes at the design stage, combining engineering logic with sustainability objectives. Similarly, Zhou et al. (2025) applied deep reinforcement learning to predict safety risks in subway construction, outperforming conventional machine learning techniques and enabling more proactive risk mitigation. These studies illustrate how AI techniques can enhance decision quality and reduce uncertainty in critical infrastructure projects. However, despite this progress, little attention has been paid to the generation of construction schedules for renewable energy plants—an equally complex task that still relies heavily on expert judgement. This study addresses that gap by applying and comparing several AI methods using real historical planning data from photovoltaic power plants. The choice to focus this study on photovoltaic plants is primarily driven by the availability of high-quality, real-world planning data, made possible by the direct involvement of co-author Jesús Gil, who brings over 12 years of experience in the development and construction of large-scale photovoltaic facilities. While the scheduling challenges faced in photovoltaic projects are comparable to those in other energy infrastructures such as wind farms or combined-cycle plants, the structured and detailed records available in the photovoltaic sector made it a suitable case study for data-driven modelling. Moreover, in practice, EPC contracts typically impose strict deadlines on the construction phase, which reinforces the need for accurate and reliable scheduling tools. It is worth noting that meteorological or regulatory variables were not included in this initial modelling phase, as their impact is typically handled through timeline renegotiations rather than influencing the original schedule baseline. Consequently, this study aims to address a pressing industry need by replacing subjective estimation with reproducible, AI-assisted planning techniques.

1.2 | Contribution

The goal of this groundbreaking study is to propose, for the first time, computational models derived from the real planning of successfully implemented large renewable power plant projects, making them a new a valuable asset for future project planning. This paper emphasises case-based models, constructed from historical examples with similar characteristics, as opposed to general models that provide information about project characteristics. To highlight the novelty of its application in this field and to explore its potential for offering innovative solutions, a comparative analysis of five machine learning-based AI techniques is conducted.

This work addresses gaps in the current literature by applying AI to real-world solar construction data and comparing the effectiveness of explainable and non-explainable models. Unlike most prior studies in construction planning, which do not use real data or focus on single-model implementation, our work provides a systematic and reproducible comparison across diverse AI techniques. The approach can be extended to other power plants or construction projects.

For this goal, the construction of large photovoltaic plants will be taken as an example and case study. This is a highly representative case in the practice of planning the construction of large renewable plant projects, given that this technology is the one that is experiencing the fastest growth, with an annual 25% in the last 10 years, and accounts for more than 30% of the current installed capacity in renewable energies (IRENA 2023). For this purpose, a dataset containing the construction planning of 25 plants is available, in which for each of them there are 216 data on their characteristics and the construction schedule. The dataset is described in Section 3.1.

The average installed power of the plants in the dataset is 53 MW, with a maximum and minimum of 185.7 and 3 MW, respectively. This dataset can therefore be considered to be representative of a set of large renewable generation plants. With this, it is intended that the algorithm for obtaining the model can be successfully replicated for other types of technologies. Furthermore, if this technique proves to be able to return good results from a dataset with a relatively small volume of projects, it could be applied to planning the construction of installations based on new renewable technologies for which there are currently not a large number of examples available and for which high future deployment is expected, for example, off-shore wind.

To this end, the work will be based on a review of the state of the art, with which the models that will be part of the comparison will be determined. These models will first be developed from a conceptual point of view. After the analysis of the dataset and its preparation, the models will be implemented in software, for training with a different data set than the one used to calculate performance metrics. The results obtained will be subject to validation and analysis, to determine the optimal model for the automatic generation of schedules.

From a formal point of view, the hypothesis of the research can be formulated as a null hypothesis in the following way: 'none of the comparative models is capable of obtaining a better prediction of a photovoltaic plant construction schedule than the predictions given by the rest of the models, nor is it capable of exceeding the baseline established by the mean value of each variable'. This study will try to prove the falsity of this hypothesis.

The specific objectives of this work are the following: (i) selection of the most suitable AI techniques for comparison, (ii) development of conceptual models, software implementation and optimization, (iii) realisation of the comparison among the models and determination of a possible optimal model, (iv) development of a tool that allows to obtain a construction schedule from input data through the optimal model.

To achieve the proposed objectives, the following steps were followed, through an empirical-quantitative methodology: (i) review of the state of the art and related works, (ii) selection of the models that will be part of the comparison, based on the analysis of the state of the art, (iii) development of the different models and metrics from a conceptual point of view, (iv) preparation and exploratory analysis of the dataset, (v) modelling and implementation in software through Python, (vi) training of the models, (vii) obtaining the metrics of the different models: MAE, MSE, R^2 and execution time, (viii) validation of the obtained results, (ix) optimization of the different models, depending on the result of the validation, comparing each one with a certain baseline (x) comparison among the models and determination of a possible optimal model, (xi) development of a tool to obtain a construction schedule from input data through the optimal model and (xii) elaboration of conclusions. The steps between training and validation will be repeated iteratively to optimise the models, until satisfactory metrics are achieved. In addition, model optimization may entail the need for further exploratory data analysis.

The rest of this paper is organised as follows: Section 2 presents the existing literature on AI techniques applied to project management with a distinction of works that use a single technique, and those who combine two or more techniques. Then, in Section 3, the experimental methodology is depicted, including the description of the case of study (i.e., the dataset), the evaluated metrics and the techniques selected for the comparison. Next, Section 4 presents the obtained results and discusses their implications. Finally, Section 5 draws the conclusions.

2 | State of the Art

There are numerous AI techniques that have been applied to project management. Within this field, the automatic elaboration of construction schedules has been the subject of research since the 1960s. Fazel Zarandi et al. (2020) present a comprehensive set of techniques used over the past 40 years, having researched more than 540 articles. Faghihi et al. (2015) review the techniques used to develop schedules, distinguishing between general models and case-based models.

From the review of the literature, the different techniques for developing schedules can be divided into two groups: basic techniques, which contemplate a single AI algorithm, and hybrid techniques, which combine several basic techniques.

2.1 | Basic Techniques

The following are the basic techniques that have been used in the development of schedules, either alone or in combination with other techniques.

Neural networks are one of the most popular of this techniques (Adeli and Karim 1997, 453; Golpayegani and Emamizadeh 2007; Rondon et al. 2008; Cheng et al. 2019). They mimic the principle of functioning of neurons in the human brain, through a set of nodes, called neurons, interconnected with each other, and exchanging information. These neurons combine into a series of layers, giving rise to the neural network. A neural network learns to interpret the data sets with which it is trained and, with this, predict behaviour in situations similar to those it has been trained on. It was in 1997 that, for the first time, this technique was applied to the elaboration of construction schedules (Adeli and Karim 1997, 453). Since then, there have been numerous advances (Golpayegani and Emamizadeh 2007; Rondon et al. 2008), including the prediction of the time needed to finish a project, with long short-term memory models (Cheng et al. 2019).

Fuzzy logic was born as a tool to describe uncertainty and inaccuracy of knowledge, imitating the way in which the human brain makes decisions in the face of uncertainty or vagueness, allowing to model complex or difficult to define systems (Cheng et al. 2012). Fuzzy logic models are composed of a 'fuzzifier', which converts inputs into fuzzy variables, a base of fuzzy rules, an inference engine from those rules, and a 'defuzzifier' that converts fuzzy variables into outputs. The fuzzy logic is expressed through a series of 'belonging functions', which allow to represent the ambiguity and, subsequently, to establish linguistic relationships between the variables. Given its limitations, due to its inability to learn from data and its dependence on expert knowledge and context, its application to project planning is carried out in conjunction with other techniques, helping to overcome the disadvantages of these.

Decision trees allow the establishment of an output, given a set of input attributes and a tree-like schema, in which one moves towards the output when making a decision in each node of the tree based on the value of the aforementioned attributes. Decision trees allow to develop simple and explainable models. The performance of an isolated tree can be improved through techniques based on a set of trees or ensemble, such as boosting and bagging. In the field of project planning, they have been used to deal with uncertainty of planning problems (Portoleau et al. 2020) as well as in obtaining priority rules (Guo et al. 2021).

Although metaheuristics are not statistical learning models in the conventional sense, their use can align with both functional definitions of supervised learning and broader conceptions of AI. From a learning perspective, these algorithms can operate on labelled data by adjusting the parameters of a predefined regression function $f(x;\theta)$ to minimise a prediction error (e.g., MSE), fulfilling the basic criteria of supervised learning as defined by Mitchell (1997). From the standpoint of AI theory, metaheuristic algorithms are recognised as intelligent agents in the sense proposed by Russell and Norvig (2020): they iteratively search for optimal solutions through rational behaviour in response to performance signals from the environment. Next, we will present metaheuristics based on evolutionary algorithms and on bioinspired emergent systems.

Evolutionary algorithms aim to simulate the evolution of species. Its principle of operation is based on the following three steps: first, it starts from a population of individuals, generated randomly. Each of them represents a possible solution to the problem. Next, the performance of each individual is evaluated as a possible solution to the problem. Finally, a new population is generated, by evolution of the pre-existing one. This process is carried out until one of the individuals meets the conditions set for the resolution of the problem (Kachitvichyanukul 2012). They have been widely applied in the area of project schedule development. Two of the algorithms with better results (Pellerin et al. 2020) are genetic algorithms and differential evolution. The former are more suitable for discrete optimization problems, while the latter are preferable for continuous optimization. In genetic algorithms, solutions are represented as chromosomes, which are evaluated and ordered from best to worst according to their value according to a fitness function. The new solutions are generated by simulating natural selection through the repeated application of three genetic operators: selection, crossing and mutation. Genetic algorithms allow a simple implementation that, directly, does not work well with complex problems, although they have been successfully incorporated into hybrid models. Differential evolution is based on an initial population of randomly generated vectors from a uniform distribution within hyperspace of possible solutions. The descent of a vector is calculated through modifications on it, obtained with the scaled difference of two other randomly selected vectors. The resulting vector only replaces its 'parent' if its performance is higher (Zhang et al. 2023). Its simple theoretical framework and ease of implementation, as well as its reliability and high performance, have made this technique widely used. The main disadvantages are its tendency to stagnation in low-quality solutions as well as to suffer from curse of dimensionality, in addition to the difficulty of adjusting hyperparameters in complex problems.

Bioinspired emergent systems are those that take as a model natural systems that together manifest properties that do not possess each of the parts of that system. There are numerous examples, as particle swarm optimization (PSO), ant colony, bee colony, firefly algorithm, shuffle frog-leaping or termite colony (Pellerin et al. 2020; Seresht et al. 2018; Tiruneh et al. 2020). PSO is possibly the most representative in the state of the art. It is inspired by the movement of systems of living organisms, such as flocks of birds. In a swarm of particles, each particle represents a possible solution, and moves to a new position by adding its velocity. Initially this position and speed are random. Once the particles reach their new position, the fitness function for each of them is evaluated and the best fitness of the particle and swarm is updated. With this information, the velocity of each particle is updated with the difference of the current position of the particle with its best historical position—its individual experience—and the difference of the current position of the particle with the best historical position of all the particles of the swarm—the group experience. This process is repeated until the stop criterion is reached. Thanks to its simplicity, its performance is good when working with large populations. One disadvantage is its tendency to return suboptimal solutions. This algorithm has been used for the successful resolution of scheduling problems (Zhang et al. 2006; Chen 2023), considering each particle as a possible schedule, with the ability to find global optimums.

In addition to the algorithms described, other basic techniques have been used within the area of automatic scheduling, such as reinforcement learning, simulated annealing or taboo list (Fazel Zarandi et al. 2020).

2.2 | Hybrid Techniques

Hybridization is the process by which two or more basic techniques are combined, to integrate their strengths and circumvent their individual weaknesses. The main lines of study integrate fuzzy logic, neural networks and evolutionary algorithms. Fuzzy systems provide explainability to solutions, but they have the difficulty of adjusting their parameters. Evolutionary algorithms allow the global optimization of these and the use of neural networks allows to optimise the local search.

Neuroevolution consists of the hybridization of neural networks with genetic algorithms, so that a genetic algorithm is responsible for the evolution of the neural network. In the first implementations, the genetic algorithm looked for the most suitable weights for the network. Subsequently, these techniques extended their scope to the determination of an optimal network topology (Stanley and Miikkulainen 2002). Algorithms based on feed-forward and fully connected networks manage to reduce the processing time for the search for an optimal network by around 80% (Harvey 2017). These techniques have been used for the resolution of planning problems (Agarwal et al. 2011) in an algorithm in which serial iterations are applied combining genetic algorithms and neural networks.

Neuro-fuzzy systems are hybrid models that combine the learning power of neural networks and the functionality of fuzzy logic. These systems usually consist of five layers of data processing: (i) input layer, where system inputs are entered, (ii) fuzzy layer, which determines the membership values of the inputs to the fuzzy sets, (iii) inference layer, which maps the fuzzy inputs to fuzzy outputs by means of a fuzzy rule system, (iv) defuzzification layer, which transforms the fuzzy outputs of the inference layer into concrete numerical values, and (v) output layer, which delivers the final output of the system. A novel approach to such techniques is the combination with deep neural networks, known in the literature as deep neuro-fuzzy systems (Talpur et al. 2023). Usual methodologies for developing neuro-fuzzy systems use the learning algorithm to obtain the fuzzification layers—for example, the shape and/or number of fuzzy membership functions—and inference—for example, the number of fuzzy rules and/or weights of the rules in the rule base. Among these techniques, the adaptive neuro-fuzzy inference system (ANFIS; Jang 1993) is the one that has achieved the best results and has been most widely used (Seresht et al. 2018; Tiruneh et al. 2020). ANFIS is a fuzzy system that adjusts the parameters of the rules with a neural network, also finding out the optimal membership functions, minimising the error between the actual output and the predicted output. It embeds learning mechanisms with two algorithms, that is, forward pass and backward pass, which usually hybridise gradient descent and a least squares estimator. An advantage of ANFIS is that it does not need expert opinion to model and train these systems (Aengchuan and Phruksaphanrat 2018). An example of the application of this technique in the field of the automatic elaboration of schedules has been the planning of work carried out by machinery (Azadeh et al. 2015). The main disadvantage of ANFIS is its computational cost, which can generate complex models for relatively simple problems and the need for a topology and membership functions for each specific application (Tiruneh et al. 2020).

The problem of automatically determining topology and membership functions has been solved by combining an evolutionary algorithm with neurofuzzy algorithms, giving rise to Evolutionary Fuzzy Neural Inference Model (EFNIM) (Ko and Cheng 2003). An evolution of EFNIM is Evolutionary Diffuse Hybrid Neuronal Network (EFHNN) (Cheng et al. 2009) which, instead of traditional neural networks, uses hybrid neural networks, with nonlinear terms, that allow to better capture nonlinearities and thus surpass the performance of traditional neural networks. Within the scope of automatic scheduling, these algorithms have been applied to the planning of photovoltaic plant projects (Gil et al. 2021). In addition to these hybrid techniques, other relevant ones are the fuzzy systems hybridised with evolutionary algorithms or with bioinspired emerging systems (Seresht et al. 2018), Evolutionary Fuzzy Support Vector Machines Inference Model (EFSIM), which follows the EFNIM scheme, although it uses support vector machine instead of neural network (Fernández Rodríguez et al. 2015), or techniques that combine adaptive reinforcement learning (Fazel Zarandi et al. 2020).

3 | Methodology

This section presents the experimental setup for the comparison of AI techniques. As a first step, the case study is described by means of the available dataset. Then, the metrics selected for the comparison are presented and, finally, the AI techniques are presented and briefly described.

3.1 | Dataset Description

As mentioned in Section 1.2, the dataset contains the project schedule for 25 selected projects. Those projects were chosen with the following selection criteria:

1. Project success: All the projects in the dataset were successfully completed, meeting capacity objectives and technical requirements. Choosing successful projects enables to train systems to generate adequate schedules.
2. Timely completion: Projects included were completed within a 10% margin of the planned timeline. With this, the system will be trained to generate schedules that fits with the usual required margins.
3. Geographical diversity: Projects come from different regions, encompassing various conditions and challenges.
4. Availability of detailed data: All the project contains detailed schedules, including tasks description, task-duration, resources, contingency plans, and so on. Each schedule includes approximately 100 tasks, covering from initial planning to grid connection.

First, an exploratory analysis of the data has been carried out. Each project contains 216 data on their characteristics and the construction schedule. A descriptive analysis of the data shows that only 85 of these variables are involved in the construction schedule of the photovoltaic plant. This 85 variables are organised as follows for each project:

- Input variables of the dataset: 23 characteristics of the plant and its planning, including data on the dimensions of the civil, mechanical and electrical works and the transmission line for connection to the grid.
- Output variables of the dataset: 62 data related to construction schedule. These data are specifically construction

task durations and intervals between the start of one task and the start of the next. They define the detailed planning of the construction of the plant, that is, from the start of the civil works to the provisional acceptance of the plant once it is in operation. This planning therefore includes the civil works, mechanical, electrical, transmission line and commissioning phases. Figure 1 shows an example of an schedule of one of the plants included in the dataset.

It is thus observed that the problem may suffer from the 'curse of dimensionality' given the high number of variables and the reduced number of examples, which causes a high distance between points in the hyperspace.

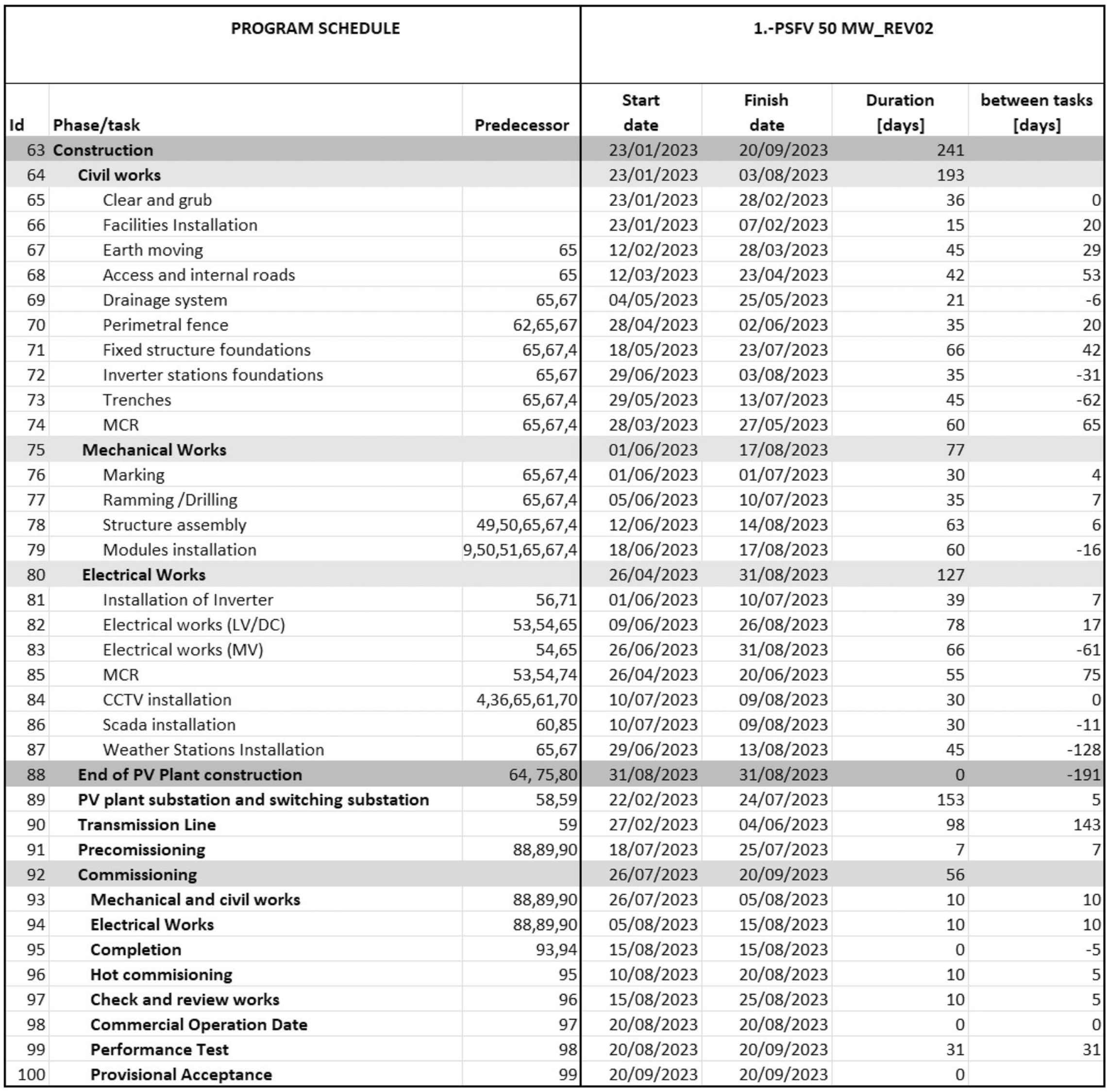
Next, a statistical-descriptive analysis of the data has been performed. This analysis has the following objectives:

1. Detect missing values.
2. Locate errors, outliers, or data out of range.
3. Analyse the behaviour of the variables and observe correlations.
4. Dimensionality reduction, to detect the possibility of eliminating features that contribute noise and affect model performance, and avoid the curse of dimensionality.

After this analysis, data debugging was carried out.

No missing values have been detected, except for the metering control room (MCR), which is not available in all the dataset schedules. Its corresponding tasks within mechanical and electrical works are identified with the Id 74 and 85 in Figure 2. The implications of this will be discussed in the final part of this subsection.

Potential outliers of a variable are those outside the interval given by 1.5 times the interquartile range between the first and third quartiles of the variable. The alternatives for debugging these outliers are (i) incorporating the input variables that may account for the outliers, or (ii) replacing the outliers by an



PROGRAM SCHEDULE			1.-PSFV 50 MW_REV02			
Id	Phase/task	Predecessor	Start date	Finish date	Duration [days]	between tasks [days]
63	**Construction**		23/01/2023	20/09/2023	241	
64	**Civil works**		23/01/2023	03/08/2023	193	
65	Clear and grub		23/01/2023	28/02/2023	36	0
66	Facilities Installation		23/01/2023	07/02/2023	15	20
67	Earth moving	65	12/02/2023	28/03/2023	45	29
68	Access and internal roads	65	12/03/2023	23/04/2023	42	53
69	Drainage system	65,67	04/05/2023	25/05/2023	21	-6
70	Perimetral fence	62,65,67	28/04/2023	02/06/2023	35	20
71	Fixed structure foundations	65,67,4	18/05/2023	23/07/2023	66	42
72	Inverter stations foundations	65,67	29/06/2023	03/08/2023	35	-31
73	Trenches	65,67,4	29/05/2023	13/07/2023	45	-62
74	MCR	65,67,4	28/03/2023	27/05/2023	60	65
75	**Mechanical Works**		01/06/2023	17/08/2023	77	
76	Marking	65,67,4	01/06/2023	01/07/2023	30	4
77	Ramming /Drilling	65,67,4	05/06/2023	10/07/2023	35	7
78	Structure assembly	49,50,65,67,4	12/06/2023	14/08/2023	63	6
79	Modules installation	9,50,51,65,67,4	18/06/2023	17/08/2023	60	-16
80	**Electrical Works**		26/04/2023	31/08/2023	127	
81	Installation of Inverter	56,71	01/06/2023	10/07/2023	39	7
82	Electrical works (LV/DC)	53,54,65	09/06/2023	26/08/2023	78	17
83	Electrical works (MV)	54,65	26/06/2023	31/08/2023	66	-61
85	MCR	53,54,74	26/04/2023	20/06/2023	55	75
84	CCTV installation	4,36,65,61,70	10/07/2023	09/08/2023	30	0
86	Scada installation	60,85	10/07/2023	09/08/2023	30	-11
87	Weather Stations Installation	65,67	29/06/2023	13/08/2023	45	-128
88	**End of PV Plant construction**	64, 75,80	31/08/2023	31/08/2023	0	-191
89	**PV plant substation and switching substation**	58,59	22/02/2023	24/07/2023	153	5
90	**Transmission Line**	59	27/02/2023	04/06/2023	98	143
91	**Precomissioning**	88,89,90	18/07/2023	25/07/2023	7	7
92	**Commissioning**		26/07/2023	20/09/2023	56	
93	**Mechanical and civil works**	88,89,90	26/07/2023	05/08/2023	10	10
94	**Electrical Works**	88,89,90	05/08/2023	15/08/2023	10	10
95	**Completion**	93,94	15/08/2023	15/08/2023	0	-5
96	**Hot commisioning**	95	10/08/2023	20/08/2023	10	5
97	**Check and review works**	96	15/08/2023	25/08/2023	10	5
98	**Commercial Operation Date**	97	20/08/2023	20/08/2023	0	0
99	**Performance Test**	98	20/08/2023	20/09/2023	31	31
100	**Provisional Acceptance**	99	20/09/2023	20/09/2023	0	

FIGURE 1 | Example of an schedule of one of the plants included in the dataset, representing the output variables.

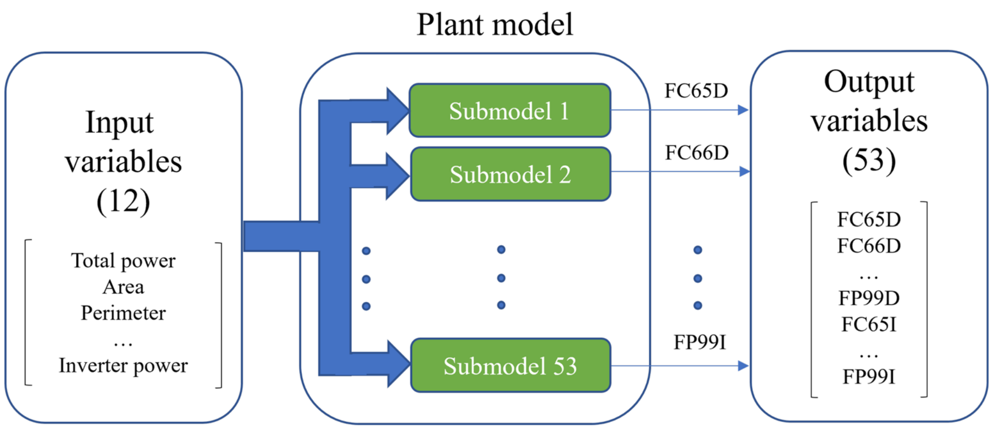


FIGURE 2 | Plant model, with a submodel for each output variable. This model is valid for trees, PSO and differential evolution.

extrapolated value. The limitations of this study when incorporating new variables imply that the alternative adopted has been the second one. For this purpose, the *k*-neighbours and linear interpolation methods—when the previous method is not robust—are used, depending on the most representative input variables, using a basic knowledge of the domain.

A study was made of the correlations between input and output variables, by groups of variables, established on the basis of expert knowledge of the problem. These groups are:

- Input variables: General variables (i.e., Total power); civil and mechanical works; electrical works; network connection.
- Output variables: Civil works duration; electrical works duration; commissioning duration; duration of the network connection.

The Pearson correlation coefficient greater than 0.5 has been established as a criterion for considering that there is a significant correlation between variables, which is equivalent to considering a $p < 0.01$. Thus, Table 1 summarises those correlations between groups of input variables and groups of output variables that can be considered significant.

The detailed analysis of input variables reveals that 10 of them are calculated, redundant or do not provide any information, so that they will be eliminated in order to favour dimensionality reduction. At this point, only 13 input variables remain in the model: general variables (power), civil works (area, perimeter, pillars, roads, ditches), mechanical works (followers, panels), electrical works (low voltage and medium voltage line, number of inverters, inverter power) and network connection (distance).

For the output of the model, that is, construction schedule, it has been sufficient to take 55 variables, related to the tasks shown in Table 2. Each task can contain two variables, each of them identified with final letter, 'D' or 'I', depending on whether it is a duration (FC65D to FP99D) or an interval between phases (FC65I to FP99I). The used notation for the output variables distinguish between the different phases with the second letter, according to Table 2, and the number corresponds to the identifier of each task, as is defined in the first column of Figure 1.

TABLE 1 | Correlations between groups of input and output variables.

Input variables group	Highly correlated output variables groups
General	Civil works duration, electrical works duration, commissioning duration
Civil and mechanical works	Civil works duration, electrical works duration, commissioning duration
Electrical works	Civil works duration, electrical works duration, commissioning duration
Network connection	Duration of construction of the network connection

Based on the detailed analysis of correlations, domain knowledge and prior modelling, two models can be established which can be assumed to be independent of each other: the network connection and the plant itself. The network model consists of only two output variables, that is, the duration of the construction of the substation and the duration of the construction of the transmission line. A linear regression from two input variables (power and distance) returns satisfactory results for this network model. The remaining problem, which will be discussed in depth below, focuses, then, on the modelling of the schedule of the plant itself, from 25 records, each containing the following data from one project:

- 12 input variables, corresponding to the characteristics of the plant described in Table 3. The distance to the existing network that can be ignored here, as it is only needed for the model of the network connection.
- 53 output variables, corresponding to the minimum durations and intervals of the tasks with which it is possible to define the construction schedule. These tasks are identified with the following Id, according to Figure 1: 65–73 (civil works), 76–79 (mechanical works), 81–84 and 86–87 (electrical works), 91 (recommissioning) and 93–99 (commissioning). It must be noticed that (i) the interval for the task 87 cannot be obtained directly from the input variables, as

TABLE 2 | Main phases and tasks in the construction of a large photovoltaic plant.

Civil works (C)	Mechanical works (M)	Electrical works (E)	Network connection (N)	Commissioning (P)
Clear and grub	Marking	Installation of inverter	Plant and switching substations	Recommissioning
Facilities installation	Ramming/drilling	Electrical works (LV/DC)	Transmission line	Mechanical and civil works
Earth moving	Structure assembly	Electrical works (MV)		Electrical works
Access and internal roads	Modules installation	Metering control room		Completion
Drainage system		CCTV installation		Hot commissioning
Perimetral fence		SCADA installation		Check and review works
Fixed structure foundations		Weather stations installation		Commercial operation date
Inverter stations foundations				Performance test
Trenches				Provisional acceptance
Metering control room				

some of the necessary predecessor tasks are not part of the model. This is because both the construction of the connection line and the substations depend on previous phases, such as the administrative processing of their projects, for which no information is available, (ii) variables related to the metering control room (Id 74 and 85) are also omitted, since, as mentioned above, they are not available in all the dataset schedules and, furthermore, a large variability has been observed, as they are asynchronous with respect to the rest of the phases, introducing a large amount of noise in the model.

3.2 | Methods and Metrics

For the validation of each model, cross-validation was setup as follows: with the total of 25 records, a battery of eight tests were carried out, each of them using a training set of 20 records (80% of the dataset) and a test set of five records (20% of the dataset). The test sets creation ensured that all existing records are part of one of the test sets.

Model fitting, consisting of obtaining of hyperparameters and training, has been performed, where applicable, with cross-validation techniques, so that the entire training set has been used to build the models. In this cases, the cross-validation in the search for hyperparameters has been independent of the cross-validation in the training, that is, nested cross-validation has been performed.

To evaluate the goodness of the different models, the 53 expected outputs for each record were computed as an average of the metrics obtained in each of the eight tests. Also, a baseline was defined as a naive model where the prediction was calculated as the average value of each of the output variables.

The comparison among the selected techniques was made using the usual metrics in the state of the art: MAE, MSE and R^2, described in Equations (1–3).

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{1}$$

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2 \tag{2}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n} \left(y_i - \bar{y}\right)^2} \tag{3}$$

where y_i represents the output obtained for the record i; $\hat{y}_i$ is the expected output for that record; $\bar{y}$ notates the baseline value (naive model) and n is the number of records.

The selection of evaluation metrics was based on their interpretability and relevance to the context of construction scheduling. MAE was chosen because it provides an intuitive measurement of the average deviation in days, which is directly meaningful for planners and stakeholders. MSE was included to emphasise the penalization of larger deviations, which are particularly critical in projects governed by strict contractual deadlines. Finally, R^2 was used to assess the proportion of variance in actual durations that is explained by the model predictions, offering a global indicator of model fit. MAPE was not prioritised, except for the total final error, where the relative error is as critical as the interpretability in

TABLE 3 | Input variables categorised by type of work.

General	Civil works	Mechanical works	Electrical works
Total power	Area	No. of trackers	Low voltage line
	Perimeter	No. of panels	Medium voltage line
	No. of foundations		No. of inverters
	Roads		Inverter power
	Trenches		

time units, and RMSE was not used as large deviations were more clearly visible with MSE.

Finally, in order to ensure the reproducibility of the study, the software environment and the main libraries, with their version, are listed below: Python 3.9.7, anfis 0.3.1, keras 2.8.0, numpy 1.20.3, pandas 1.3.4, pyswarms 1.3.0, scikit-learn 0.24.2, scikit-fuzzy 0.4.2, scipy 1.7.1, tensorflow 2.8.0, XGBoost 1.6.1.

3.3 | Selected AI Techniques

From the application of the state of the art to the problem under study, and following criteria of optimality, explainability, variety and novelty, five algorithms have been selected for the comparative performance evaluation, guided by their suitability for structured project planning data and their complementary characteristics.

1. Decision trees, for their high explainability and ability to provide explicit decision rules, making them valuable for understanding the influence of individual input variables on scheduling outcomes. Random forest and XGBoost were also included for their strong predictive performance and robustness, especially in the presence of noisy or redundant features, though at the cost of reduced interpretability.
2. PSO, for having achieved some of the best results in Pellerin et al. (2020). In addition, it is possible to make an explainable implementation.
3. Differential evolution, as a representative of evolutionary algorithms, having achieved the best results in Pellerin et al. (2020). It can also be implemented in an explainable way.
4. Neuroevolution, because of its ability to explore complex, nonlinear search spaces, and its novelty solving this problem.
5. ANFIS, as a neuro-fuzzy model that balances learning capacity and explainability thanks to the fuzzy layer, particularly suitable for small to medium-sized datasets. It shows also a good performance in a wide variety of problems.

The inclusion of metaheuristics as PSO and differential evolution in this study reflects their established role in the soft computing branch of AI, and allows for a broader methodological comparison alongside machine learning models.

Next, the techniques used in the comparison are developed.

3.3.1 | Decision Trees

For this technique, both an isolated tree and two ensemble algorithms will be used: XGBoost and Random forest. The implementation of the algorithm is done using a regression tree for each of the output variables (Testas 2023). Each tree (submodel) is fed with all input variables, as shown in Figure 2.

This strategy provides sufficient flexibility and independence to the output variables, which is of great importance given the small number of records. The optimal combination of hyperparameters (maximum depth and minimum number of elements per leaf) per k-fold cross validation is sought, with $k = 5$. In order to avoid overfitting, the maximum depth of each tree is limited to six levels and a number of examples per leaf of 2.

Regarding to the results, there is a clear improvement over the baseline in all metrics (see Table 4). The improvement obtained can be easily understood through the mean absolute error (MAE) achieved, of 5.57 days in test, compared to 9.17 in the baseline. There is also a clear improvement in MSE and R^2. As for the computational cost of obtaining the model, it is small, with an average of 25.93 s for the execution of the algorithm. The existence of overfitting to the training data can be observed. The MAE in training is only 2.38 days, and the coefficient of determination R^2 reaches, in training, a very high value (0.8748, compared to 0.3650 in test).

To try to improve its performance, tests are carried out with a model based on boosting—XGBoost—and another based on bagging—Random Forest. These models will lose explainability, as they are constituted by a set of trees.

3.3.2 | XGBoost

This technique consists of developing a model formed by successive regression trees, giving in each of the successive iterations a greater weight to those examples that have been incorrectly classified previously. The resulting model is the sum of all calculated trees (Brownlee 2016). This strategy allows to solve problems detected with the extreme values of the output variables. In order to evaluate the goodness of the predictions during training, and thus perform the readjustment, it is necessary to define a validation set within the training set it. Five records have been taken for this purpose. With this single caveat, the training process is analogous to that described for decision trees.

TABLE 4 | Comparison of the results obtained by the different techniques.

		Base line	Regression tree	XGBoost	Random forest	PSO	Differential evolution	Neuro evolution	ANFIS
MAE train	days	8.48	2.38	1.80	0.06	7.71	8.48	4.34	2.10
MSE train	days^2	259.12	32.89	46.74	0.96	105.28	259.13	49.41	20.35
R^2 train	—	0.0	0.8748	0.8166	0.9963	0.6150	0.0	0.8079	0.9242
MAE test	days	9.17	5.57	5.38	5.26	9.80	9.19	5.36	4.54
MSE test	days^2	275.86	129.29	117.45	137.65	193.34	276.16	75.41	67.19
R^2 test	—	−0.4201	0.3650	0.4354	0.3225	0.0383	−0.4216	0.5909	0.6545
Time	s	0	26	157	47	572	213	704	2106

The results obtained by this model are slightly better in all metrics than those of an isolated regression tree (MAE of 5.38 days vs. 5.57, see Table 4). The overfitting is also slightly lower, as can be deduced from the fact of a higher MSE in the training data, which implies that the model performs better even allowing for larger extreme deviations from the training data.

3.3.3 | Random Forest

This algorithm is based on the generation of a set of decision trees. For each of them, only a part of the input variables is taken (usually $\sqrt{p}$, being p the number of input variables) and a part of the training set (usually $2/3$). The result of the prediction is taken as the average of the result of each of the individual trees. This is intended to avoid the overfitting that usually exists when modelling with a single tree, at the cost of losing a simple explainability, being 'diluted' among all models. After various tests, the optimal number of trees has been set at 500. A greater number of trees leads to greater overfitting, without improving test metrics. Fewer trees lead to stability problems in solutions.

In terms of the results obtained, there is a slight improvement over XGBoost in MAE (5.26 vs. 5.38), although both MSE and R^2 degrade (see Table 4). This implies that the Random forest-based model, while having a good average performance, is more susceptible to give solutions that differ in an extreme way—thus increasing its MSE—and is worse at explaining variances—given its lower coefficient of determination-. It is observed that the models resulting from this technique fit the training data very precisely, which in practice translates into an overfitting that prevents efficient inference on the test data.

To summarise, decision tree techniques assign an average or median value to output variables based on certain ranges in the input variables. Therefore, they will produce a systematic bias in the prediction. These techniques do not allow for adequate modelling of some of the output variables, especially the intervals between starting dates, since in these, the samples are too far apart in some of the output variables of the hyperspace. Splitting into a large number of leaves produces overfitting, and the alternative, with a small number of leaves, produces overfitting. On the contrary, splitting into a small number of leaves assigns the same value to a set of values in the same relatively large range of inputs.

The conceptual answer to the above problem can be found in this and the following optimisation techniques. All of them allow continuous output variables to be obtained. This means that the output variables are obtained from the input variables through parametric functions. The problem will therefore be to obtain these parameters.

3.3.4 | PSO

The model used for PSO uses a swarm for each of the output variables (Figure 1) (Miranda 2018). Each of the particles i in a swarm represents a possible solution and seeks the optimal position in hyperspace. Each particle moves at instant t to a new position $x_{i,t}$, adding its velocity $V_{i,t}$ to its previous position $x_{i,t-1}$:

$$x_{i,t} = x_{i,t-1} + V_{i,t} \tag{4}$$

Once the particles reach their new position, the fitness function for each particle is evaluated and the best fitness of the particle and the swarm is updated. With this information, the velocity of each particle $V_{i,t}$ is updated as can be seen in Equation (5), as the sum of:

- The previous velocity $V_{i,t-1}$, weighted by an inertia factor w. The larger this factor is, the more a global search will be favoured. A small value will favour a local search.
- The difference of the particle's position from its best historical position $x_{i,best}$. This difference is weighted by a 'cognitive parameter' c_1, as well as by a random factor r_1 between 0 and 1. This sum represents the individual knowledge of the particle.
- The difference of the particle's position with the best historical position of all particles in the swarm $x_{global,best}$. This difference is weighted by a 'social parameter' c_2, as well as by a random factor r_2 between 0 and 1. This summand endows the particle with knowledge based on group experience.

This process is repeated until the stopping criterion is reached.

$$V_{i,t} = wV_{i,t-1} + c_1 r_1 \left(x_{i,best} - x_{i,t-1} \right) + c_2 r_2 \left(x_{global,best} - x_{i,t-1} \right) \quad (5)$$

The dimensions of the position of a particle represent the parameters that relate the corresponding output variable to the input variables. Linear dependencies between input variables and output variables have been assumed. Thus, the estimation of each output variable y_i can be represented as a function of the n input variables x_j as:

$$\hat{y}_i = \sum_{j=1}^{n} \left(w_j x_j \right) + b_i \quad (6)$$

being w_j, b_i the parameters to be calculated, which define the position of the particle: a linear parameter w_j for each of the 12 input variables plus an independent term b_i. The cost function, to be minimised, is the sum of the squares of the errors between the calculated value $\hat{y}_i$ of the output variable and its actual value y_i for the m samples of the training set.

$$cost\left(y_i, \hat{y}_i \right) = \frac{1}{m} \sum_{i=1}^{m} \parallel \left(y_i - \hat{y}_i \right) \parallel^2 \quad (7)$$

The cost function given by Equation (7) is evaluated for each of the particles in the swarm. With it, it is possible to update the position of each particle, considering that it will be attracted by its best historical position, with a weight c_1, the best historical position of the swarm, with a weight c_2, and inertia against its previous motion, with weight w. The best results have been obtained with a total of 1000 particles per swarm. The network hyperparameter search strategy is the one that achieves the best results, around $c_1 = 0.8$, $c_2 = 0.3$ and $w = 0.8$. This search is carried out for each of the output variables. The algorithm converges very quickly.

Nevertheless, the results returned, shown in Table 4, are not good. In the case of MAE (9.80 days), it is even lower than that given by the baseline (9.17 days). The poor performance of this algorithm is due to a known problem: its tendency to converge to sub-optimal solutions. While the average predictions are not good, it does produce an improvement over the baseline in terms of outliers (see the MSE) and in its ability to explain variations in the output (see the MSE). To explain variations in output (given by R^2). One of the solutions proposed to solve suboptimal convergence problems is the use of local optimisation techniques, in which a number of ‘sub-swarms’ are defined. This prevents a single swarm from falling into a local minimum. However, tests with this local algorithm have shown very poor performance.

3.3.5 | Differential Evolution

This algorithm uses a population of vectors for each output (Figure 1) and can be approached through different strategies. In its original definition (Neri and Tirronen 2010), for each vector x_i of the initial population, three vectors x_r, x_s and x_t are drawn pseudo-randomly. With these three vectors, a new vector x'_{desc} is calculated, which allows to generate offspring.

$$x'_{desc} = x_t + F\left(x_r - x_s \right) \quad (8)$$

F is a mutation factor, which controls the length of the scan vector $\left(x_r - x_s \right)$ and thus defines the distance from x_i to its offspring. F can take values between 0 and 1 or even slightly greater than 1. In its most common implementation, the vector x_t is usually the one that, within the whole population, offers the best value of a fitness function. Once the provisional offspring x'_{desc} has been generated, each ‘gene’ (component) of this vector can be exchanged with the corresponding gene of x_i, as follows:

$$x_{desc,j} = \begin{cases} x'_{desc,j}, \text{ if } rand(0,1) < C_R \\ x_{i,j}, \text{otherwise} \end{cases} \quad (9)$$

where j is the index of the gene under consideration, $rand(0,1)$ is a random number between 0 and 1, and C_R is a parameter known as the recombination constant. The offspring replaces the previous generation if its fitness is superior.

The strategy that has given the best results is known as ‘best1bin’ (Virtanen et al. 2020). In it, the difference between the two randomly selected vectors is used to mutate the best member of the population. This mutated vector will be the one that transfers some of its parameters to the vector to evolve, depending on the recombination constant. The cost function, to be minimised, is analogous to that described in PSO and thus given by Equation (7).

It is needed to define also the following basic hyperparameters: population size, mutation rate and recombination rate. The best results have been obtained with a population of 25 vectors, a degree of mutation between 0.5 and 1 and a degree of recombination of 0.5. The best results returned by this algorithm are very close to the mean value of each variable and, therefore, the metrics are very similar to those given by the baseline, as can be seen in Table 4. As for the execution time (212 s on average), it is not excessive, although it is useless as it does not provide any value on the baseline. To try to improve its performance, larger regions of hyperspace have been explored, increasing population and mutation and reducing recombination. These tests have returned lower yields than those given by the baseline. those given by the baseline. The ‘rand1bin’ strategy, which uses a random vector instead of the ‘best vector’, has also been employed, without outperforming the results given by ‘best1bin’. In short, this algorithm suffers in this case from the problem already described in the state of the art, consisting in the stagnation of the results given by ‘best1bin’. art, consisting of stagnation in solutions lower than those that can be qualified as suboptimal.

3.3.6 | Neuroevolution

The implementation carried out in this technique seeks to define the topology of the neural network through the genetic algorithm. It starts from an input layer with a number of neurons equal to the total number of input variables (12) and an output layer with a number of neurons equal to the total number of output variables (53). While NEAT-based approaches are interesting from a theoretical point of view, their practical implementation is not attractive. Thus, a Keras-based implementation has been chosen (Harvey 2017). In it, each individual of the population is a forward-powered, fully connected neural network with several hidden layers. Each individual is defined by the following parameters or 'genes', for which the final values are:

1. Number of neurons per hidden layer. 8 predefined values in a list of values between 120 and 370 neurons.
2. Number of hidden layers. 2, 3 or 4 layers.
3. Activation function: ReLU or eLU.
4. Optimizer: Adam, Adagrad, Adadelta, RMSprop.

As hyperparameters of the genetic algorithm, the following are defined:

1. Number of generations: 20.
2. Population of each generation: 20 networks per generation.
3. Ratio of the population that must remain after each generation: 0.4.
4. Random selection: The probability that a network not selected for its fitness will continue in the population. A value of 0.1 is used.
5. Degree of mutation: represents the probability that a network will undergo some mutation. A value of 0.2 is used.

The training of each network is executed with 100 epoch with early stop condition in case after 10 iterations there are no improvements in performance. Once the selection has been carried out by the genetic algorithm, the model obtained will be the individual (network) with the best fitness of the last generation. The fitness function must be maximised, so it has been defined as the MSE changed sign. The best solutions are usually found with 2 or 3 layers and around 240 neurons per layer. Figure 3 shows an example of architecture. The activation function in the hidden layers that has returned better results has been ReLU. Each layer, except the output layer, then carries a dropout of 0.2, to regularise the behaviour of the network. Finally, the output layer also incorporates ReLU activation function.

The results obtained are presented in Table 4. It can be seen that the performance obtained is much higher than that established by the baseline. The substantial improvement of this method is not observed in the MAE (5.36 days), but in the reduction of the value of the MSE (75.41), which means a better behaviour in the face of possible large errors, and a relevant increase in the coefficient of determination R^2, very close to 0.6. The average running time is 703 s. This may seem high but, as indicated in Section 2.2, this strategy reduces the time used by brute force algorithms by 80%.

3.3.7 | ANFIS

In this implementation (Meggs 2023), computing resources effectively limit the number of input variables to a maximum of five per model. This has implied the need to reformulate each output variable according to a maximum of five input variables, as shown in Figure 4.

Figure 5 details how the decomposition of each set of ANFIS models is performed, using as an example the ANFIS electrical work set. It is observed how for this case there are 11 ANFIS models (one per output variable), each of them fed by the same 4 input variables.

In turn, each of these ANFIS models implements a neurodiffuse system, exemplified in Figure 6 for the variable FE81D (duration of the laying task of the low voltage line).

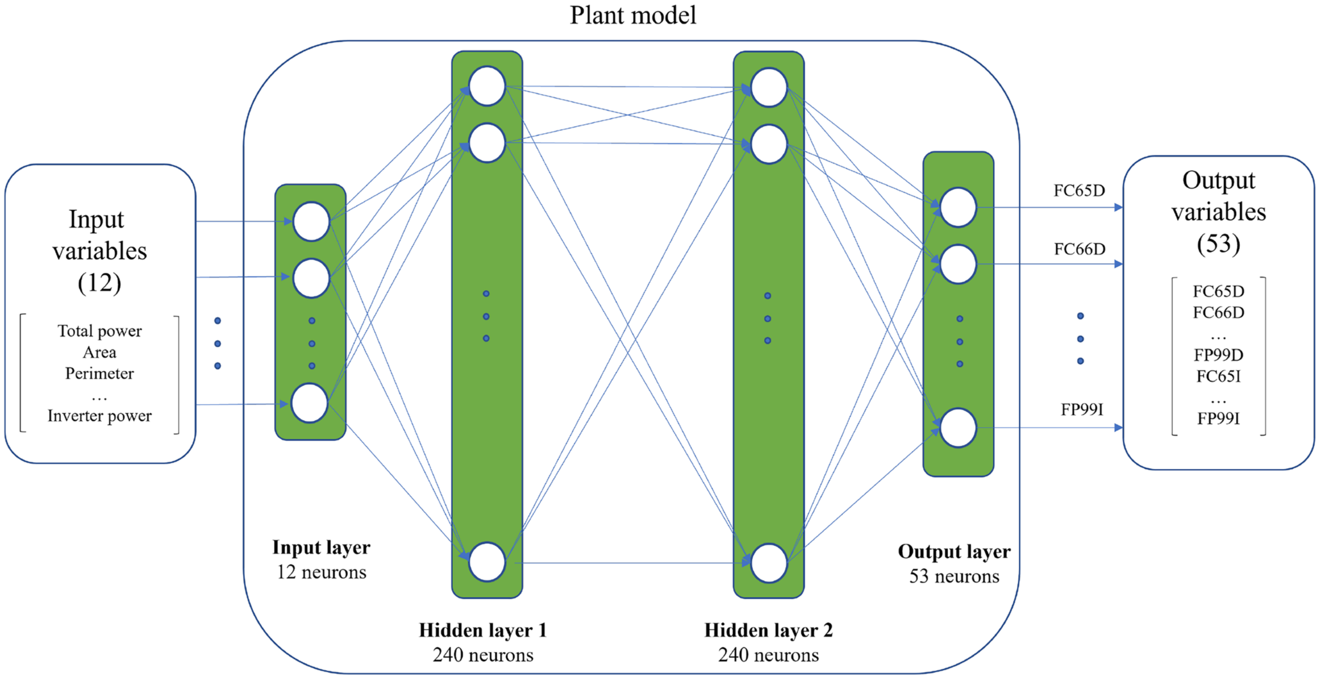


FIGURE 3 | Neural network architecture obtained by the genetic algorithm.

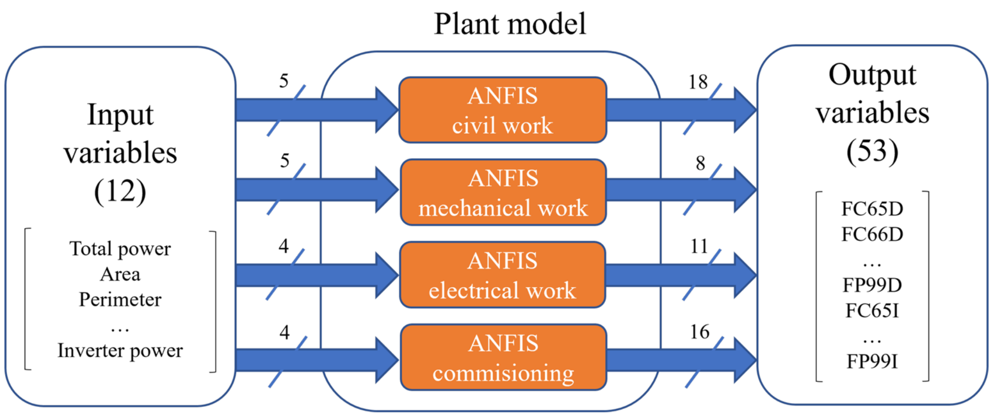


FIGURE 4 | Global model of the plant based on ANFIS.

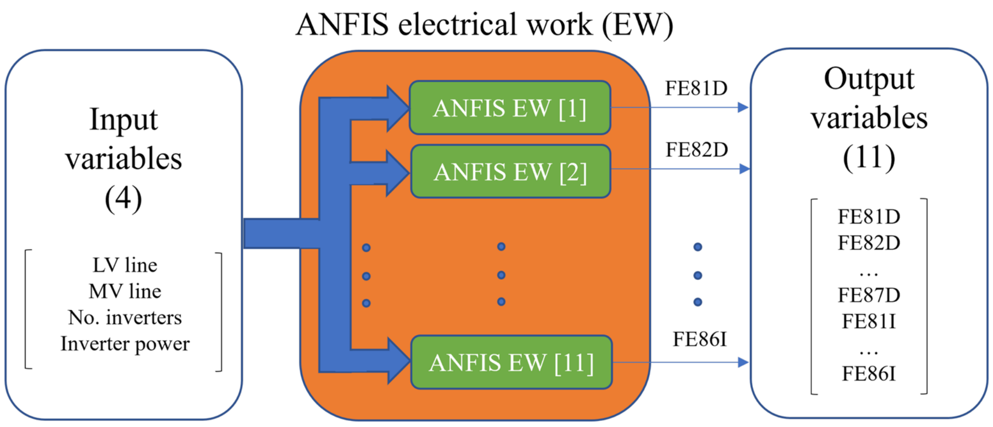


FIGURE 5 | ANFIS model of the group of electrical work variables.

ANFIS requires the definition of an initial set of fuzzy membership functions $\mu(x)$ for each of the input variables x. A definition that has returned good results are with three Gaussian membership functions per variable, with $\sigma = 0.45$ and $\mu = 0$, 0.5 and 1. ANFIS performs several epochs in training. Figure 7 shows the adjustment of the prediction to the 20 plants of the training set after 4 epochs. It is noted that a number of epochs higher than this does not necessarily lead to better results and in some cases shows instability.

The results obtained with ANFIS are shown in Table 4. The average performance is in all metrics much higher than the baseline. It also improves the performance of the other algorithms on the test set, not having incurred a significant degree of overfitting. The only disadvantage of ANFIS is its long running time, which on average takes more than 35 min.

4 | Results and Discussion

4.1 | Results

Numerical results are presented at Table 4, and the main metrics obtained on average in the tests are displayed graphically in Figure 8. It can be seen that the techniques that perform better are, in this order:

1. ANFIS.
2. Neuroevolution.
3. Decision trees.
4. PSO.
5. Differential evolution.

ANFIS stands out for its good performance in all metrics, followed by neuroevolution. Decision trees present a close result of MAE, even though performance in MSE and R^2 is inferior. The next model in terms of performance is PSO. Although MSE and R^2 improves the baseline, its performance in MAE is worse, which means that, in practice, it is not predictively useful. Finally, the results of differential evolution are practically analogous to those given by the baseline, that is, by the mean.

The comparison of training metrics versus test metrics allow to evaluate possible overfittings, which is preserved to a greater degree in tree techniques, especially in Random forest, which in training returns MAE close to 0 and R^2 close to 1. In the rest of the techniques there is no significant degree of overfitting, although the results of ANFIS in training are clearly superior to those of the test.

Another relevant metric is runtime. Tree-based techniques are the fastest (between 26 and 157 s). PSO and differential evolution have average execution times of 572 and 213 s. The algorithms with the longest execution time are neuroevolution (704 s) and, especially, ANFIS (2176 s). The analysis of the results for each of the eight sets of tests shows consistency in

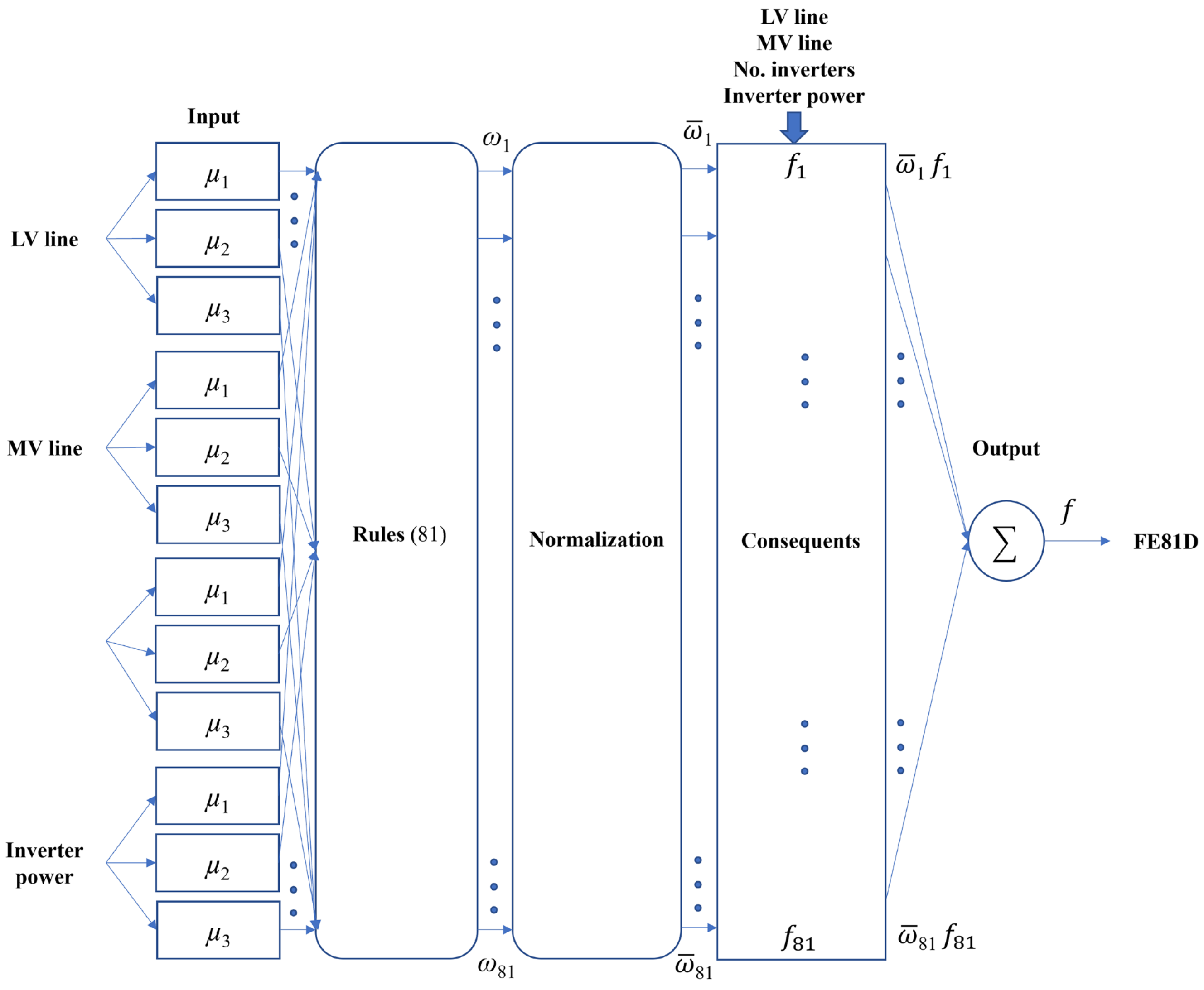


FIGURE 6 | Detail of the neurodiffuse system of an ANFIS model.

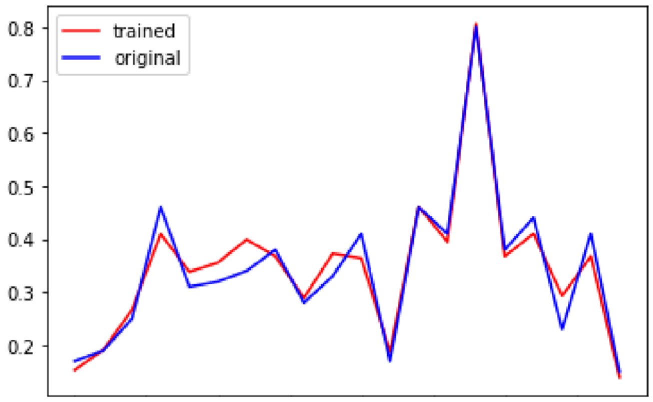


FIGURE 7 | Adjustment of the ANFIS model to the training set.

the order of performance of the algorithms, with ANFIS being the technique that offers the most balanced performance in all tests.

4.2 | Discussion

The results presented in the previous section are now analysed and interpreted in light of existing literature, algorithmic behaviour and its underlying causes, and practical implications for construction planning in solar plants.

4.2.1 | Interpretation of Algorithm Performance

The technique that has obtained the best performance in error metrics, consistently and without relevant overfittings, has been ANFIS. Membership functions and fuzzy rules manage to properly assess the importance of each of the inputs and return a tight output. This good performance has its counterpart in a high computational cost, with an average of 35 min, three times higher than that of the next most expensive technique. Its explainability is not immediate, since each output is composed of the weighted result of 3^n rules, where n is the number of input variables. A simplification of the rule set can mitigate these disadvantages.

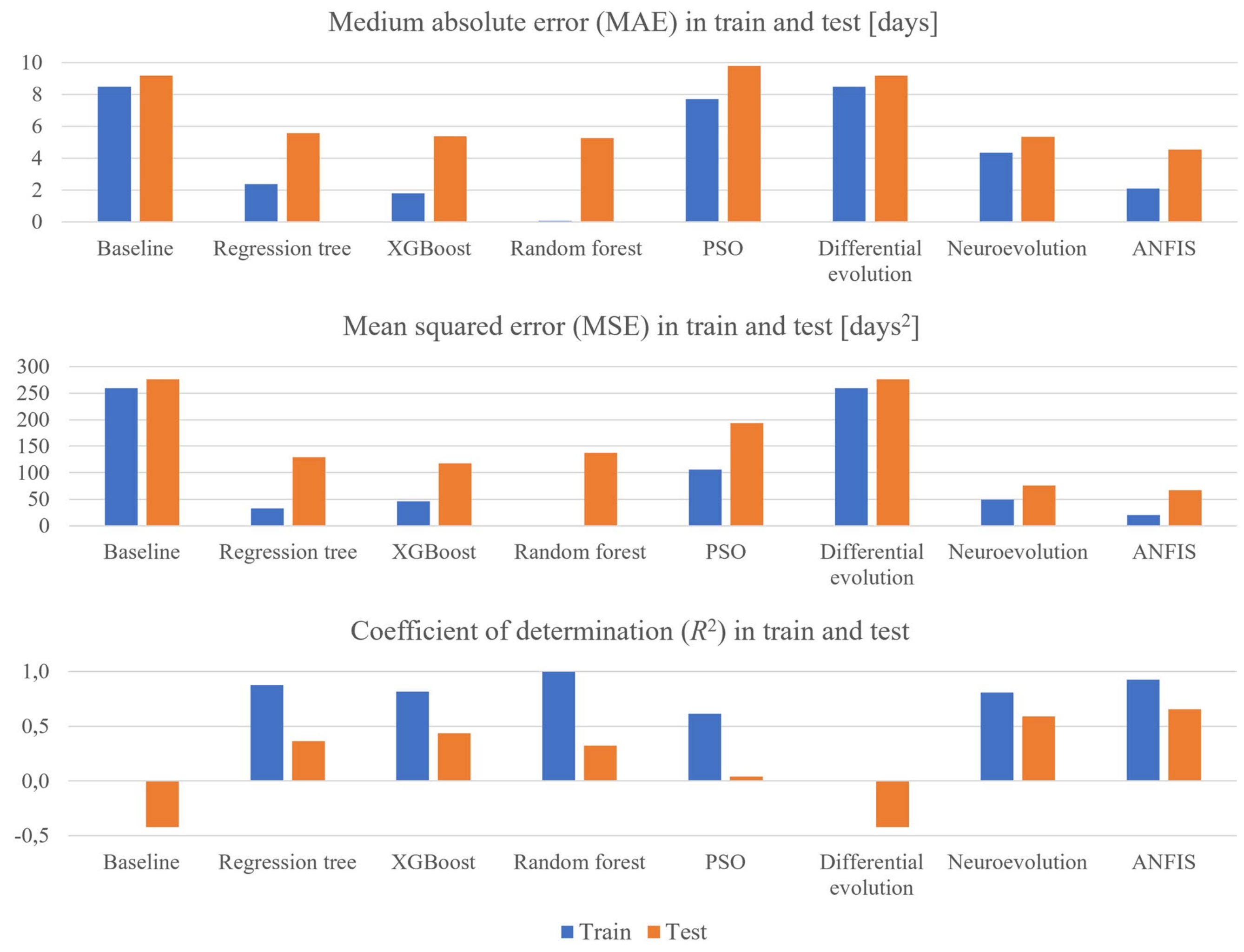


FIGURE 8 | Graphical summary of the results of the different techniques.

The next technique in performance is neuroevolution. In relation to ANFIS, it's MSE and R^2 are 10% lower, and 20% in the case of MAE. These good results are repeated in all tests and represent a significant reduction in computational cost compared to ANFIS. Another advantage of this method is that its tendency to overfit is the least. Its main disadvantage is its lack of explainability, with typically over 70,000 parameters, with no physical significance. At a considerable computational cost (average of 12 min), its solutions are lower than those of ANFIS.

As a result, hybrid techniques perform better than basic techniques. Within the basic techniques, those that have offered better results are those based on decision trees. Its classification in order of performance in test is: XGBoost, Random forest and isolated regression tree, without very significant differences. Its MAE is very similar to that of neuroevolution, although the results in and MSE are clearly lower, which R^2 means that these techniques present a worse behaviour in terms of large errors, a lower explainability of the model and the existence of possible biases. Despite this, the result is reasonably good to obtain a first approximation, with fast execution times. These techniques, however, suffer from a significant overfitting. Finally, only the technique based on an isolated tree offers good explainability.

The results obtained with PSO have been worse than those given by decision trees and can be described as unsatisfactory. MSE and R^2 improve the baseline. However, MAE, with a value close to 10 days, offers a performance below the baseline. This technique quickly converges to generally suboptimal solutions. Its computational cost has been relatively high (around 10 min). An eventual advantage, if it has achieved more satisfactory results, is its trend to overfitting. Although the linearity hypothesis has not been an adequate solution to this problem, it allows the model to be explainable. Higher quality solutions, as has been observed, introduce greater complexity into their modelling.

The technique that has returned worse results in the comparative has been differential evolution. The result of the prediction given by this model is very approximately the mean value of the variable in the training set, so the error metrics are very similar to those of the baseline. One reason for the poor quality of the predictions is the linearity hypothesis adopted between the input and output variables. This hypothesis prevents exploring the set of best solutions, so the algorithm stagnates in suboptimal regions. Its execution time is low (only higher than decision trees), which would be an eventual advantage if the solutions were satisfactory.

When comparing the scalability of the algorithms, Regression tree and Random Forest tend to be more scalable due to their relatively simpler structure, allowing them to handle large datasets efficiently. XGBoost, while highly accurate and robust, requires more computational resources and careful tuning, impacting scalability to some extent. PSO and Differential Evolution generally scale less efficiently with increasing problem size due to their iterative nature and need for population-based search, which can be computationally intensive. Neuroevolution, which evolves neural networks, also faces scalability challenges, particularly with complex architectures and large datasets. Finally, ANFIS is in theory less scalable due to the intricate interplay between its components, requiring substantial computational power as the problem size grows.

4.2.2 | Practical Applications

An additional perspective in the analysis of results is its contribution in absolute terms, beyond the comparative, for its practical application. For this, the model that has returned the best results, ANFIS, is used. A first measure of the goodness of the results achieved can be taken by contrasting the value of MAE in test (4.54 days) against the average absolute value of the output variables (29.17 days). This represents an average relative error of 16% in the calculation of the durations and intervals between the different tasks of construction. On the other hand, a R^2 of 0.6545 can be considered very well against a very small and highly varied data set. For reference, the baseline given by the mean of the training data obtains a R^2 of -0.4201.

The final test of the goodness of the model consists of comparing real test schedules against the schedules that the model would elaborate. For the maximum verisimilitude of the test, the ANFIS model obtained with that test has been used—among the eight carried out—where the error metrics have been the most similar to the average metrics. The results obtained with this model are remarkably good. Observing the total duration D of the schedules obtained (Table 5), their relative error e is 8%, with a maximum of 10%, without significant upward or downward biases. It can be concluded that the errors existing in each of the individual tasks (16%) do not contain relevant biases, which allows their partial mutual cancellation, to reach the aforementioned error in the total duration of 8%.

From a practical standpoint, the 8% average error in total project duration translates to a deviation of approximately 24 days in an average project of 300-day. This level of accuracy is substantially better than what is usually obtained through heuristic or expert-based planning. The difference in performance among the models can be attributed to several factors. ANFIS and neuroevolution are better suited to capture non-linear relationships in the data. Tree-based methods, while computationally efficient, tended to overfit the training data. PSO and DE, although inherently flexible, suffered from stagnation in local optima and performed worse under the linear approximation constraint applied in this study. The limited size of the dataset may have favoured algorithms capable of learning with fewer examples and robust to noise. In terms of practical deployment, ANFIS offers a compelling balance between accuracy and interpretability. The rules and membership functions could be inspected and validated by human experts, making it suitable for industrial applications that require transparency. By contrast, neuroevolution and tree ensembles, although performant, operate as black boxes and may be unsuitable for contexts requiring traceability or regulatory compliance.

For implementation, the model obtained with ANFIS has been used, trained from the entire available data set (25 records), in order to improve its predictive capacity. This model makes it possible to generate construction schedules for photovoltaic plants in a very simple way, from a set of only 14 input variables. As a result, task durations and intervals between tasks are obtained, as shown in Figure 9.

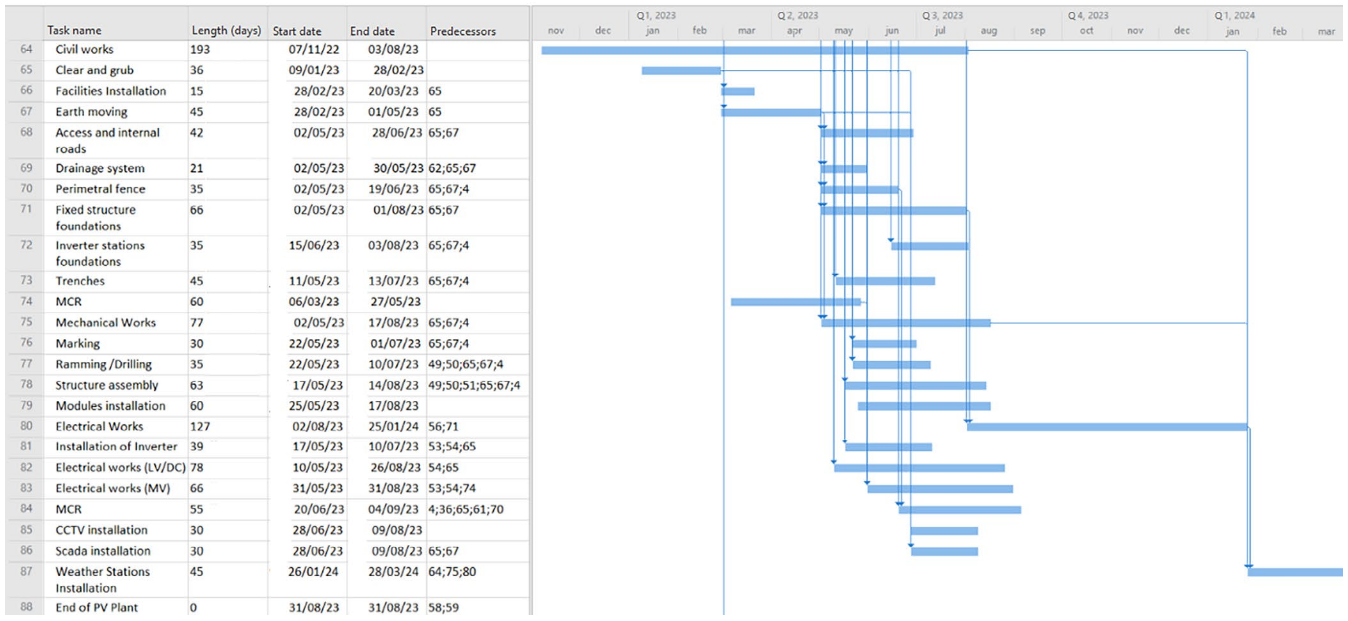
In addition, there are practical implications for those who may lack access to advanced computational tools. Commercial AutoML platforms such as IBM Watson AutoAI, Google Vertex AI, Amazon SageMaker Autopilot, and Microsoft Azure ML Studio allow users to train and deploy some of the analysed models—including decision trees, ensemble methods, and even metaheuristic approaches like PSO—without the need for coding. These platforms provide graphical interfaces and automated pipelines, significantly lowering the barrier to entry for adopting AI in project planning. Additionally, ANFIS predictions can be embedded into lightweight applications, such as Excel macros or web interfaces, allowing small-scale project managers to benefit from advanced scheduling predictions without extensive technical knowledge.

5 | Conclusions and Future Work

Meeting the energy transition objectives requires increasing the implementation pace of large renewable plants. The main factor determining the success of these projects is proper planning. This serves as motivation to try to solve, through AI techniques, the elaboration of adequate planning. A tool capable

TABLE 5 | Results of the schedules obtained with ANFIS.

		Plant 1	**Plant 21**	**Plant 11**	**Plant 3**	**Plant 5**
		96 MW	**146 MW**	**65 MW**	**50 MW**	**37 MW**
D real	days	253	340	220	226	162
D predicted	days	278	311	242	215	175
e total	days	25	−29	22	−11	−13
e relative	%	10%	9%	10%	5%	8%



	Task name	Length (days)	Start date	End date	Predecessors
64	Civil works	193	07/11/22	03/08/23	
65	Clear and grub	36	09/01/23	28/02/23	
66	Facilities Installation	15	28/02/23	20/03/23	65
67	Earth moving	45	28/02/23	01/05/23	65
68	Access and internal roads	42	02/05/23	28/06/23	65;67
69	Drainage system	21	02/05/23	30/05/23	62;65;67
70	Perimetral fence	35	02/05/23	19/06/23	65;67;4
71	Fixed structure foundations	66	02/05/23	01/08/23	65;67
72	Inverter stations foundations	35	15/06/23	03/08/23	65;67;4
73	Trenches	45	11/05/23	13/07/23	65;67;4
74	MCR	60	06/03/23	27/05/23	
75	Mechanical Works	77	02/05/23	17/08/23	65;67;4
76	Marking	30	22/05/23	01/07/23	65;67;4
77	Ramming /Drilling	35	22/05/23	10/07/23	49;50;65;67;4
78	Structure assembly	63	17/05/23	14/08/23	49;50;51;65;67;4
79	Modules installation	60	25/05/23	17/08/23	
80	Electrical Works	127	02/08/23	25/01/24	56;71
81	Installation of Inverter	39	17/05/23	10/07/23	53;54;65
82	Electrical works (LV/DC)	78	10/05/23	26/08/23	54;65
83	Electrical works (MV)	66	31/05/23	31/08/23	53;54;74
84	MCR	55	20/06/23	04/09/23	4;36;65;61;70
85	CCTV installation	30	28/06/23	09/08/23	
86	Scada installation	30	28/06/23	09/08/23	65;67
87	Weather Stations Installation	45	26/01/24	28/03/24	64;75;80
88	End of PV Plant	0	31/08/23	31/08/23	58;59

FIGURE 9 | Example of a schedule of one of the plants obtained with ANFIS.

of generating schedules automatically, without the influence of human bias, can be of great help when establishing initial reference plans. This work adopts an innovative approach, as executes for the first time five machine learning-based algorithms over actual planning of successfully executed projects. The compared algorithms include three basic and two hybrid techniques. The real schedules correspond to large photovoltaic plants, which can be considered a representative case of use in the field of large renewable plants.

The results obtained in the comparison show a superior performance of hybrid techniques. The one that has returned the best results is ANFIS, with an absolute average error of around 4.5 days and a coefficient of determination R^2 of 0.65. Obtaining a model with a higher result than the rest and better than the baseline has also made it possible to discard the null hypothesis formulated in Section 1.2. The neuroevolution algorithm has achieved satisfactory results, slightly lower than those of ANFIS, with a lower computational cost. As for the basic techniques, only techniques based on decision trees can clearly improve the prediction given by the average of the training set. Both PSO and differential evolution have suffered from problems of stagnation in suboptimal solutions, the latter being unable to overcome the baseline.

Finally, a tool has been developed, capable of establishing reference plans based on the model and from 14 input variables—one of which is the start date of construction. The results obtained by this tool, with a relative error of 8% in the total duration of planning, have exceeded previous expectations.

Some possible future works that have arisen as a result of this study are:

1. Testing new hybrid models, in view of the superior result achieved by this type of models. EFHNN is part of ongoing research lines, although simpler models such as EFNIM or those based on NEAT can achieve satisfactory results. As a consequence of the promising decision trees, a possible line of future research would be to hybridise them with models that allow the setting of a continuous output and decrease their high overfitting. Particle swarm optimization and differential evolution models could also benefit from their hybridization, in order to explore models of higher complexity than linear ones. Finally, a simplification of the ANFIS rule base according to expert criteria would make it possible to include a greater number of input variables for each output variable. For this type of test, we also propose the use of specific computing systems, with greater capacity than the one used for the development of this work.
2. Developing basic models that allow overcoming the limitations of those who have been part of the comparison. Support Vector Machine is an algorithm suitable for problems such as the one considered, with a small number of registers and a large number of variables. These properties have included it in current lines of research. Bioinspired models mentioned in the state of the art, as ant colony and, especially, shuffled frog-leaping, could overcome the tendency to fall into suboptimal solutions that has been observed in the algorithms tested in this typology.
3. Performing parametric modelling or with nonlinear dependencies. This line of research requires an expert knowledge of the domain, in order to establish the necessary parameters (e.g., powers or products of the input variables).
4. Use not only expert knowledge but also AI techniques to simplify the number of ANFIS rules in order to facilitate the explainability of the model.

5. Another promising direction is the training of large language or generative models (e.g., LLaMA 3, Mistral) using structured datasets from energy infrastructure projects, enabling AI systems to support planning justification, adaptive rescheduling or document generation through natural language interfaces.
6. Enriching the input dataset to fine-tune the model. This line of research consists of different aspects:
   - A greater number of records, which allows for improved inference.
   - A greater diversity in the data of the input variables, to improve inference as well. The current data set can only predict schedules given a certain level of work resources. Another constraint is that all the plants are connected to a 132 kV voltage network, preventing inference at different voltages. In addition, 9 of the 25 plans are for similar installations.
   - Incorporate additional variables to explain apparently contradictory values, that is, why a construction interval is very different between two plants, when the characteristics that determine it are the same (e.g., the construction time of the perimeter enclosure in two plants with the same perimeter). This line of work is complemented by a thorough debugging and homogenization of the data, to be carried out by experts.
7. Adjusting the duration of tasks depending on the time of year. Factors such as hours of daylight or weather can significantly affect the progress of work.
8. Extending the model to other technologies, as could be onshore and offshore wind farms.
9. Extending the model to other processes which require planning, as could be budgeting, procurement, logistics, workforce allocation, and cost-risk estimation.
10. Although this study focused on the construction planning of utility-scale photovoltaic plants, the proposed methodology has broader applicability. The modelling framework and AI techniques used here can be extended to other renewable energy infrastructures such as wind farms, as well as to conventional energy projects like combined-cycle or nuclear power plants. Additionally, the approach may be adapted to other large-scale construction domains—including transportation infrastructure (e.g., highways, railways, ports) and industrial facilities—where planning relies on milestone-based scheduling and faces similar constraints in terms of cost, deadlines, and coordination of resources.

Thus, this work contributes to the literature by providing a robust framework for applying AI to scheduling with real industrial data, opening a path for both advanced and explainable algorithms.

Data Availability Statement

The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

References

Adeli, H., and A. Karim. 1997. “Scheduling/Cost Optimization and Neural Dynamics Model for Construction.” *Journal of Construction Engineering and Management* 123, no. 4: 450–458.

Aengchuan, P., and B. Phruksaphanrat. 2018. “Comparison of Fuzzy Inference System (FIS), FIS With Artificial Neural Networks (FIS+ANN) and FIS With Adaptive Neuro-Fuzzy Inference System (FIS+ANFIS) for Inventory Control.” *Journal of Intelligent Manufacturing* 29: 905–923.

Agarwal, A., S. Colak, and S. Erenguc. 2011. “A Neurogenetic Approach for the Resource-Constrained Project Scheduling Problem.” *Computers & Operations Research* 38, no. 1: 44–50.

Agyekum-Mensah, G., and A. D. Knight. 2017. “The Professionals' Perspective on the Causes of Project Delay in the Construction Industry.” *Engineering Construction and Architectural Management* 24, no. 5: 828–841. https://doi.org/10.1108/ECAM-03-2016-0085.

Azadeh, A., N. Hosseini, S. Abdolhossein Zadeh, and F. Jalalvand. 2015. “A Hybrid Computer Simulation-Adaptive Neuro-Fuzzy Inference System Algorithm for Optimization of Dispatching Rule Selection in Job Shop Scheduling Problems Under Uncertainty.” *International Journal of Advanced Manufacturing Technology* 79: 135–145.

Brownlee, J. 2016. XGBoost with Python: Gradient Boosted Trees With XGBoost and Scikit-learn.

Chen, M. 2023. “Scientific and Technological Innovation Rapid Emergency Resource Constraint-Improved Particle Swarm Optimization Project Scheduling Method.” *Journal of Advanced Manufacturing Systems* 22, no. 1: 165–180.

Cheng, M.-Y., Y.-H. Chang, and D. Korir. 2019. “Novel Approach to Estimating Schedule to Completion in Construction Projects Using Sequence and Nonsequence Learning.” *Journal of Construction Engineering and Management* 145, no. 11: 04019072.

Cheng, M.-Y., L.-C. Lien, H.-C. Tsai, and P.-H. Chen. 2012. “Artificial Intelligence Approaches to Dynamic Project Success Assessment Taxonomic.” *Life Science Journal* 9, no. 4: 5156–5163.

Cheng, M.-Y., H.-C. Tsai, and E. Sudjono. 2009. “Evolutionary Fuzzy Hybrid Neural Network for Conceptual Cost 2009 Proceedings of the 26th ISARC Estimates in Construction Projects.” In *26th International Symposium on Automation and Robotics in Construction (isarc2009)*, 512–519.

Faghihi, V., A. Nejat, K. F. Reinschmidt, and J. H. Kang. 2015. “Automation in Construction Scheduling: A Review of the Literature.” *International Journal of Advanced Manufacturing Technology* 81: 1845–1856.

Fazel Zarandi, M. H., A. A. Sadat Asl, S. Sotudian, and O. Castillo. 2020. “A State of the Art Review of Intelligent Scheduling.” *Artificial Intelligence Review* 53: 501–593.

Gil, J., J. M. Torres, and R. González-Crespo. 2021. “The Application of Artificial Intelligence in Project Management Research: A Review.” *International Journal of Interactive Multimedia and Artificial Intelligence* 6, no. 6: 54–66.

Golpayegani, S. A. H., and B. Emamizadeh. 2007. “Designing Work Breakdown Structures Using Modular Neural Networks.” *Decision Support Systems* 44, no. 1: 202–222.

Gong, Q., J. Wu, Z. Jiang, M. Hu, J. Chen, and Z. Cao. 2024. “An Integrated Design Method for Remanufacturing Scheme Considering Carbon Emission and Customer Demands.” *Journal of Cleaner Production* 476: 143681. https://doi.org/10.1016/j.jclepro.2024.143681.

Guo, W., M. Vanhoucke, J. Coelho, and J. Luo. 2021. “Automatic Detection of the Best Performing Priority Rule for the Resource-Constrained Project Scheduling Problem.” *Expert Systems With Applications* 167: 114116.

Harvey, M. 2017. "Let's Evolve A Neural Network With a Genetic Algorithm," Coastline Automation.

IRENA. 2023. *Renewable Energy Statistics 2023*. International Renewable Energy Agency.

Jang, J.-S. 1993. "ANFIS: Adaptive-Network-Based Fuzzy Inference System." *IEEE Transactions on Systems, Man, and Cybernetics* 23, no. 3: 665–685.

Kachitvichyanukul, V. 2012. "Comparison of Three Evolutionary Algorithms: GA, PSO, and DE." *Industrial Engineering and Management Systems* 11, no. 3: 215–223.

Ko, C.-H., and M.-Y. Cheng. 2003. "Hybrid Use of AI Techniques in Developing Construction Management Tools." *Automation in Construction* 12, no. 3: 271–281.

Meggs, T. 2023. "ANFIS: Python implementation of an Adaptive Neuro Fuzzy Inference System." GitHub. https://github.com/twmeggs/anfis.

Miranda, L. J. 2018. "PySwarms: A Research Toolkit for Particle Swarm Optimization in Python." *Journal of Open Source Software* 3, no. 21: 433.

Mitchell, T. M. 1997. *Machine Learning*. McGraw-Hill.

Neri, F., and V. Tirronen. 2010. "Recent Advances in Differential Evolution: A Survey and Experimental Analysis." *Artificial Intelligence Review* 33, no. 1: 61–106. https://doi.org/10.1007/s10462-009-9137-2.

Pellerin, R., N. Perrier, and F. Berthaut. 2020. "A Survey of Hybrid Metaheuristics for the Resource-Constrained Project Scheduling Problem." *European Journal of Operational Research* 280, no. 2: 395–416.

Portoleau, T., C. Artigues, and R. Guillaume. 2020. "Robust Predictive-Reactive Scheduling: An Information-Based Decision Tree Model." In *Information Processing and Management of Uncertainty in Knowledge-Based Systems: 18th International Conference, Ipmu 2020, Lisbon, Portugal, June 15–19, 2020, Proceedings, Part Iii 18*, 479–492. Springer International Publishing.

Regona, M., T. Yigitcanlar, B. Xia, and R. Y. M. Li. 2022. "Opportunities and Adoption Challenges of AI in the Construction Industry: A PRISMA Review." *Journal of Open Innovation: Technology, Market, and Complexity* 8, no. 1: 45.

Rondon, R. L. A., A. S. da Carvalho, and G. I. Hernández. 2008. "Neural Network Modelling and Simulation of the Scheduling." In *International Conference on Information Technology for Balanced Automation Systems*, 231–238. Springer.

Russell, S., and P. Norvig. 2020. *Artificial Intelligence: A Modern Approach*. Pearson Education.

Seresht, N. G., R. Lourenzutti, A. Salah, and A. R. Fayek. 2018. "Overview of Fuzzy Hybrid Techniques in Construction Engineering and Management." In *Fuzzy Hybrid Computing in Construction Engineering and Management: Theory and Applications*, 37–107. Emerald Publishing Limited.

Stanley, K. O., and R. Miikkulainen. 2002. "Evolving Neural Networks Through Augmenting Topologies." *Evolutionary Computation* 10, no. 2: 99–127.

Talpur, N., S. J. Abdulkadir, H. Alhussian, M. H. Hasan, N. Aziz, and A. Bamhdi. 2023. "Deep Neuro-Fuzzy System Application Trends, Challenges, and Future Perspectives: A Systematic Survey." *Artificial Intelligence Review* 56, no. 2: 865–913.

Testas, A. 2023. "Decision Tree Regression With Pandas, Scikit-Learn, and PySpark." In *Distributed Machine Learning With Pyspark: Migrating Effortlessly From Pandas and Scikit-Learn*, 75–113. Springer.

Tiruneh, G. G., A. R. Fayek, and V. Sumati. 2020. "Neuro-Fuzzy Systems in Construction Engineering and Management Research." *Automation in Construction* 119: 103348.

Virtanen, P., R. Gommers, T. E. Oliphant, et al. 2020. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python." *Nature Methods* 17, no. 3: 261–272.

Zhang, H., H. Li, and C. Tam. 2006. "Particle Swarm Optimization for Resource-Constrained Project Scheduling." *International Journal of Project Management* 24, no. 1: 83–92.

Zhang, Y., G. Dai, L. Peng, and M. Wang. 2023. "Enhancing Differential Evolution Algorithm Through a Population Size Adaptation Strategy." *Natural Computing* 22, no. 2: 379–392.

Zhou, Z., W. Zhuo, J. Cui, H. Luan, Y. Chen, and D. Lin. 2025. "Developing a Deep Reinforcement Learning Model for Safety Risk Prediction at Subway Construction Sites." *Reliability Engineering and System Safety* 257: 110885. https://doi.org/10.1016/j.ress.2025.110885.