steel research international

steel research international

RESEARCH ARTICLE OPEN ACCESS

A Novel Approach to Energy Management in Electric Steelworks

Valentina Colla[1] | Stefano Dettori[1] | Silvia Cateni[1] | Marco Vannucci[1] | Antonella Vignali[1] | Claudio Mocci[1] | Irene Dovichi[1] | Davide Chionna[1] | Lorenzo Vannini[1] | Enrico Paluzzano[2] | Costanzo Pietrosanti[2] | Davide Onesti[2] | Daniele Venier[2]

[1]TeCIP Istitute, Scuola Superiore Sant'Anna, Pisa, Italy | [2]Danieli Automation S.p.A, Buttrio, Italy

Correspondence: Lorenzo Vannini (lorenzo.vannini@santannapisa.it)

Received: 1 October 2025 | **Revised:** 17 January 2026 | **Accepted:** 26 February 2026

Keywords: electric arc furnace | electric steelmaking | energy management system | ladle furnace | machine learning | optimization

ABSTRACT

Electric steelworks are a paradigmatic representation of the concept of circular economy, as it recycles steel components at the end-of-life products. Moreover, its importance is foreseen to grow according to the increasing demand of decarbonizing steel production to meet the ambitious goals of the European Green Deal. The electric arc furnace-based route is still characterized by a limited diversification of energy supply sources thus, managing the three factors of smart energy management, energy prices, and production planning can be jointly considered as a crucial leverage for reducing production costs while ensuring satisfaction of energy demand coming from the different processes and developing digital approaches and tools to implement the fast adaptation to power grid behaviour. The article describes a novel energy management system based on innovative components and a flexible infrastructure, which uses machine learning and an optimization approach to minimize electricity consumption and level trends by matching intelligent production planning and power grid offer and related energy costs. The developed solution and the set of neural networks-based models estimating electricity consumption in electric arc furnace and ladle furnace based on production information are described. The models were trained and validated using production and process data from a real steelworks.

1 | Introduction

The electric steelmaking route is nowadays considered as the key melting technology for the decarbonization of the steel ector due to its pillars such as the circular economy, the direct reduction of iron, the flexibility to manage the energy demand, and so on.

The electric arc furnace (EAF) route implements the concept of circular economy since decades, far before it gained its current popularity because of its related standard practice to use end-of-life steel scrap which represents the primary raw material of the EAF, to be currently enriched by the direct reduced iron (DRI).

Regarding the management of electricity, it pushes the implementation of energy produced by renewable sources, the development of hydrogen production and use, and, despite a prudent mention, also the new declinations of nuclear plants in the form of small nuclear reactors. Therefore, the holistic combination of these factors can effectively lead to reduce the environmental impact compared to the integrated cycle [1].

In addition, it is characterized by a very reduced inertia toward the operational flexibility, as the EAF-based production can be stopped or slowed down in a relatively fast and easy way, enabling adaptation to the nowadays strongly variable market demand.

However, electric steelmaking also shows critical issues that are still only partially resolved, such as the dependence on scrap

quality [2, 3], which is currently quite variable with the consequence of impacting on both product quality and electricity consumption. In addition, the electricity consumption of EAF is huge, which poses challenges in terms of cost, availability, and energy diversification in a market that is increasingly sensitive to ever-changing social and geo-political conditions

Moreover, the unavoidable transformation of the power market in terms of production technologies and energy sources, the possibility to match power demand and offer becomes crucial because of the foreseen renewable' increase in the global power offer and so, it can be expected an increase of variability depending on daily up to seasonal variations. Figure 1 shows the global electric energy production per source typology in 2023.

A major point is represented by the substitution of roughly 18.500 TWh with green energy non only considering the mere numerical value but also the potential grid behaviour, in particular in developed countries. Renewables are not stable, the transition toward hydrogen is very capital intensive and far, also considering the timespan fixed by the EU. It implies that the 'workhorse' of the power generation, which notoriously are characterized to be almost stable in terms of oscillations, might be rapidly replaced. Consequently, the need to deal with a less stable grid behaviour must be considered.

Moreover, it is necessary to strongly push toward enhanced precision of demand forecasting and achievement of the optimal match between demand and offer. This means also that the cooperation between energy producers and large users is necessary, as confirmed by the current demand of higher EAF power. In such a context, intelligent planning solutions to reduce and stabilize electricity consumption play a fundamental role for the economic and environmental sustainability of the process [4].

In general, solving a process scheduling problem by considering the condition of the energy market is a rather daunting task for operators, especially when economic sustainability must be balanced by compliance with normal plant operating constraints and production pace. Reducing the cost of energy supply implies leveling consumption within a daily time horizon, limiting peaks, or planning the most energy-intensive production batches in periods of lower electricity prices. Therefore, in the last decades numerous researchers investigated this topic through different approaches. For instance, discrete optimization approaches to address energy-related constraints were proposed in [5, 6] and later in [7], where a resource task network (RTN)-based formulation was adopted to incorporate the EAFs' flexibilities to reduce the electricity cost. In contrast, a continuous-time mixed integer linear programing approach to minimize the energy consumption was proposed in [8] and further refined in other works [8–11], while a mixed integer nonlinear programing was proposed in [12]. However, these works assume fixed and constant energetic demand of each process, which is a very strong hypothesis. An attempt to overcome this limitation was made by Fiorani et al. [13], who exploited very simplified models to estimate energy consumption of the different processes that do not consider the variability of consumption with respect to the different steel grades and some process parameters. More recently Li et al. [14] proposed a new coordinated optimization framework to integrate the process-level production scheduling into the bidding model for EAF steelmaking, which achieves effective coordination between production and bidding in both energy and reserve ancillary service markets to improve the trading profit of the steel mill that can sell the excess flexibility in electricity markets to get additional financial compensation. This work exploits a RTN-based production scheduling model with piecewise constant estimates of energy consumption of the different process, which are, however, not easily customizable, as they depend on a series of constant parameters lacking automatic tuning procedures based on industrial data, not easily interpretable by operator, and not considering consumption variability based on the steel grade and the main process parameters. A similar approach showing analogous drawbacks is followed by Su et al. [15].

In contrast, smart production scheduling requires tools that can estimate energy consumption as a function of the steel grade to produce, the availability of scrap, the production recipe and the

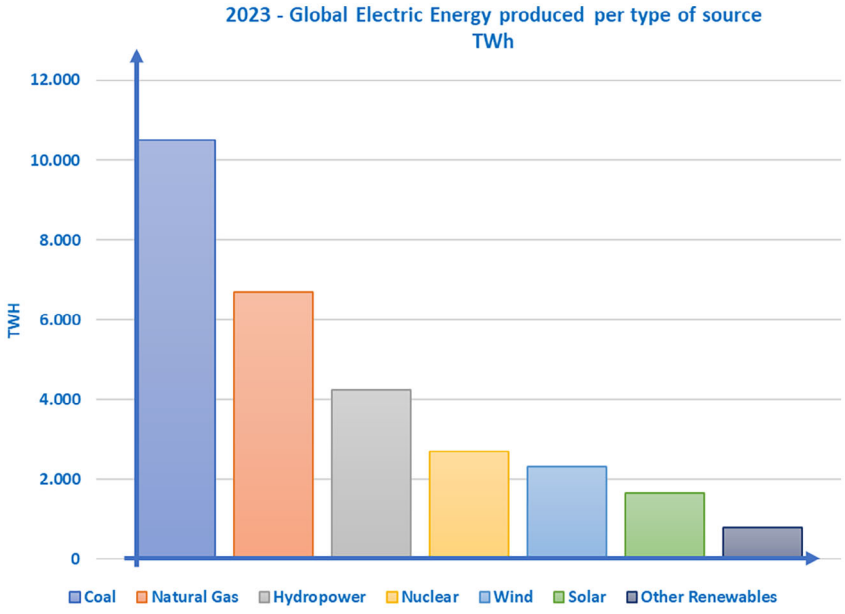

FIGURE 1 | Global production of electric energy in 2023 per typology of source.

main process measurements or information [16]. These tools and models must also be easy to customize and balance the accuracy of numerical predictions and their computational cost, thus numerical models based on complex equations are ineffective to get practical indications inline. In this context, machine learning (ML) can support modeling of energy-intensive production processes, thanks to its ability to exploit large volumes of data and to numerically approximate chemical/physical behaviours of complex systems. Indeed, an appropriate data collection at each energy intensive process is a fundamental preliminary step for the viability of artificial intelligence (AI) and ML-based solutions, such as highlighted in a relevant methodological analysis proposed by Kahlid et al. [17].

In the past, relevant research efforts have been spent on modeling the EAF process and determining its energy consumption via different approaches, given its predominant role in energetic demand of the connected steelmaking route and the increasing availability of process data and computational resources experienced in the last decades thanks to the unprecedent progresses of monitoring systems and ICT supported new developments. The beginning of the third millennium was still dominated by physics-based and mechanistic models, such as the interesting investigation pursued by Camdalı and Tunc on the influence of different process parameters on the EAF electric energy consumption [18], or the mathematical model of Logar et at [19]. designed for control and optimization purposes. However, these models are quite difficult to adapt and customize to different steelworks and production, therefore, an ever-growing interest has emerged toward data-driven and ML-based modeling of electric energy consumption at the EAF [20–22], including deep learning (DL) [23, 24]. In such a context, Zadnowik et al. [25] moved a further step forward by investigating time series forecast of EAF energy usage, employing various methods including long-short term memories for different forecasting horizons and achieving also a quite fine time resolution. Recently Lu et al. [26] proposed the use of Hybrid AI to merge mechanistic models and AI-based approaches targeting improvement of accuracy and interpretability of the model outcomes. The goal of developing interpretable ML models of EAF electric consumption is also at the core of the work of Carlsson et al. [27], who explored different approaches. However, while most works exploiting ML and DL emphasize the need for accurate data filtering to remove unreliable data and ensure reliable model performance, they rarely treat in depth the selection of input variables for such models, which is mostly left to the experience of the designer and based on process knowledge. A noticeable result in this context is achieved by Behera et al. [28], who propose a hybrid framework integrating feed-forward neural networks (FFNN) and genetic programing (GP) to estimate the EAF specific energy consumption using actual industrial data, as for GP models a selection of input variables is automatically made. However, their analysis is restricted to the specific case of a DRI-based EAF and targets the optimization of the EAF process itself, rather than plant-wide reduction of energy cost. Another interesting example is provided by García-Nieto et al. [29], who followed a structured variable selection approach to develop different ML-based models for energy consumption by integrating a bio-inspired optimization approach with multivariate adaptive regression splines. However, their work focuses on the overall electric energy consumption instead of targeting specific processes composing the production chain, which is a requirement if the goal is to adapt process schedule to variable energy cost along the day, and refers to integrated steelworks.

In contrast, concerning the Ladle Furnace (LF) process, conventional parametric models are applied since decades to estimate energy consumption [30], but they often do not consider the specificity of each steel grade, while ML-based models and other AI approaches have been applied so far to estimate of forecast the temperature of the molten steel [31–35] or end point contents of relevant components, such as C [36] and S [37].

Within the project entitled 'Energy Management in the Era of Industry 4.0' (EnerMIND) a prototypical solution was developed to support plant managers and operators in optimizing the production schedule to reduce the energy production costs by exploiting data-driven models to estimate the energy consumption and duration of the main processes of the electric steelmaking route. The ambition of the prototype was to develop a solution overcoming what is already available in terms of flexibility, simplicity of use and customization, and meeting the current demands of existing steelworks, that is, requiring low capital investments, moderate skills of the operators and being adaptable to all regimes of production (from intense to medium low). The prototype should also be open to future evolutions following the ongoing digitalization process of the sector: although process data and computational resources are surely increasingly available due to the technological evolution, especially medium-low size EAF-based facilities are proceeding at very different paces. Therefore, the prototype must be initially based on information that is normally available in most companies and capable to demonstrate economic advantages that pay back the limited efforts required to set it up. However, it should also be easy to evolve as soon as the IT landscapes evolves as well to include more sophisticated models and optimization algorithms.

The paper introduces the modeling approach and the results obtained during the prototyping of the ML-based models of EAF and LF, which were trained and validated through data from one real steelworks. This work aims to fill an existing gap related to models estimating the energy consumption of EAF and LF (i.e. the most energy-intensive processes of the electric steelmaking route), which are simple, computationally efficient, easy to customize also thanks to codified procedures for data filtering and reduction, and capable to provide a rough albeit reliable estimate of the process energy demand as a function of some relevant and usually available process parameters for each steel grade also in process conditions that have not been directly tested in the past without the need to install additional sensing and measuring devices and/or to drastically modify the existing IT system.

Moreover, the paper describes how the models are used in the context of a novel optimization approach to reduce the energy cost in electric steelworks by adjusting the production schedule based on the variable price of energy. Also for the optimization, a meta-heuristic approach is proposed according to the same inspiring principles, namely simplicity and interpretability, to gain the company's trust, but also transferability and flexibility for future introduction of more sophisticated algorithms the conditions at the company enable this step.

2 | Experimental Section

2.1 | The Available Data

In the present work, ML is used to estimate the following parameters characterizing consumptions of the EAF and LF processes: average power consumption, energy consumption, power-on time and heat time (i.e. total process duration). Any data-driven model, including neural networks (NN), is strongly affected by the quality of the data used for their design. Beyond quality, data format and volume are the first requirement to be taken into consideration during the modeling phase. In the present study, the used data are organized on a heat basis, and their information content describes the material flows, processing steps and energy use along the production chain, from scrap charge in the EAF, to melting, secondary metallurgy and continuous casting (CC).

For the models related to the EAF process, the available potential input data and their basic statistics, such as minimum/maximum value (min/max), mean value (mean) and standard deviation (std) are presented in Table 1. For those connected to the LF process, the available dataset includes only a small set of features (see Table 2).

These event-driven data had to be preliminarily analyzed to detect missing data, and to identify unreliable data (outliers), that are often due to manual entries or sensor faults by both analyzing data distribution [38] and exploiting personnel experience. Variables with null variance are irrelevant, thus a total number of 19 variables were considered.

Data cleaning included removal of heats where at least one variable that is considered important by operators was missing (to ensure complete samples for supervised learning), and addressing missing/altered records due to transmission issues or manual alterations, through a reconciliation of process timestamps and sequence consistency checks.

Regarding outliers, a first screening could be performed through comparison with basic statistics (average value μ and Standard Deviation σ) complemented by distribution-based checks to flag suspicious values (e.g., inconsistent tapping temperatures or abnormal heat-time durations). For instance, variables values differing by more than 3σ from the average value are labeled as suspicious, but this criterion is fully valid if the distribution of the considered variable is Gaussian, an hypothesis that can be too 'strong' such as depicted in Figure 2, which shows some exemplary histograms related to 4 potential variables related to the EAF process: among these 4 variables, the data distribution can be considered Gaussian only for Oxygen and Carbon total consumptions (Figures 2b,c). The situation is analogous for the other variables. It is therefore important also to consider

TABLE 1 | Potential input variables for the model of the EAF process.

No	Variable name, unit	Min	Max	Mean	Std
1	Steel Grade	–	–	–	–
2	Gas total consumption [Nm^3]	63	818	410.86	116.12
3	Oxygen total consumption [Nm^3]	1468	5363	2423.1	294.34
4	Carbon total consumption [Kg]	100	4487	903.68	323.68
5	Loaded scrap weight [ton]	67.26	83.33	75.46	2.33
6	Coal loaded in briquettes [ton]	0	0	0	0
7	Sheet type scrap loaded [ton]	0	44.25	12.59	9.86
8	Corrective type scrap loaded[ton]	0	13.6	0.53	1.85
9	Special demolitions 8 M [ton]	0	49.72	20.41	9.54
10	Demolition scrap 6 mm [ton]	0	0	0	0
11	Shredded scraps [ton]	0	23.11	4.86	4.62
12	Recovery Ingots type scrap	0	6.15	0.70	1.23
13	Light demolition scraps[ton]	0	0	0	0
14	Turnings scrap type [ton]	0	30.16	4.89	5.99
15	Dross scrap type [ton]	0	10.61	2.71	2.35
16	Returns CD [ton]	0	23.43	1.34	3.47
17	Returns NCD [ton]	0	8.91	0.11	0.61
18	Cast iron-based scrap [ton]	0	17.21	6.21	3.77
19	Cast iron [ton]	0	0	0	0
20	E103 scrap type [ton]	0	56.16	9.56	8.68
21	E202 Thick new production scraps [ton]	0	0	0	0
22	Loose deep drawing scrap type [ton]	0	41.44	8.62	8.54
23	DRI [ton]	0	17.01	2.91	4.53
24	EAF ferro-alloys [ton]	0.046	2.35	0.83	0.37

TABLE 2 | Input variables for the model of the LF process.

No	Variable name, unit	Min	Max	Mean	Std
1	Start weight [ton]	57.05	80.88	70.43	3.92
2	First sample temperature [°C]	0	1733	1298	639.9
3	Target steel grade	–	–	–	–
4	Total weight of LF Alloys [ton]	0.038	2.621	0.5513	0.2598

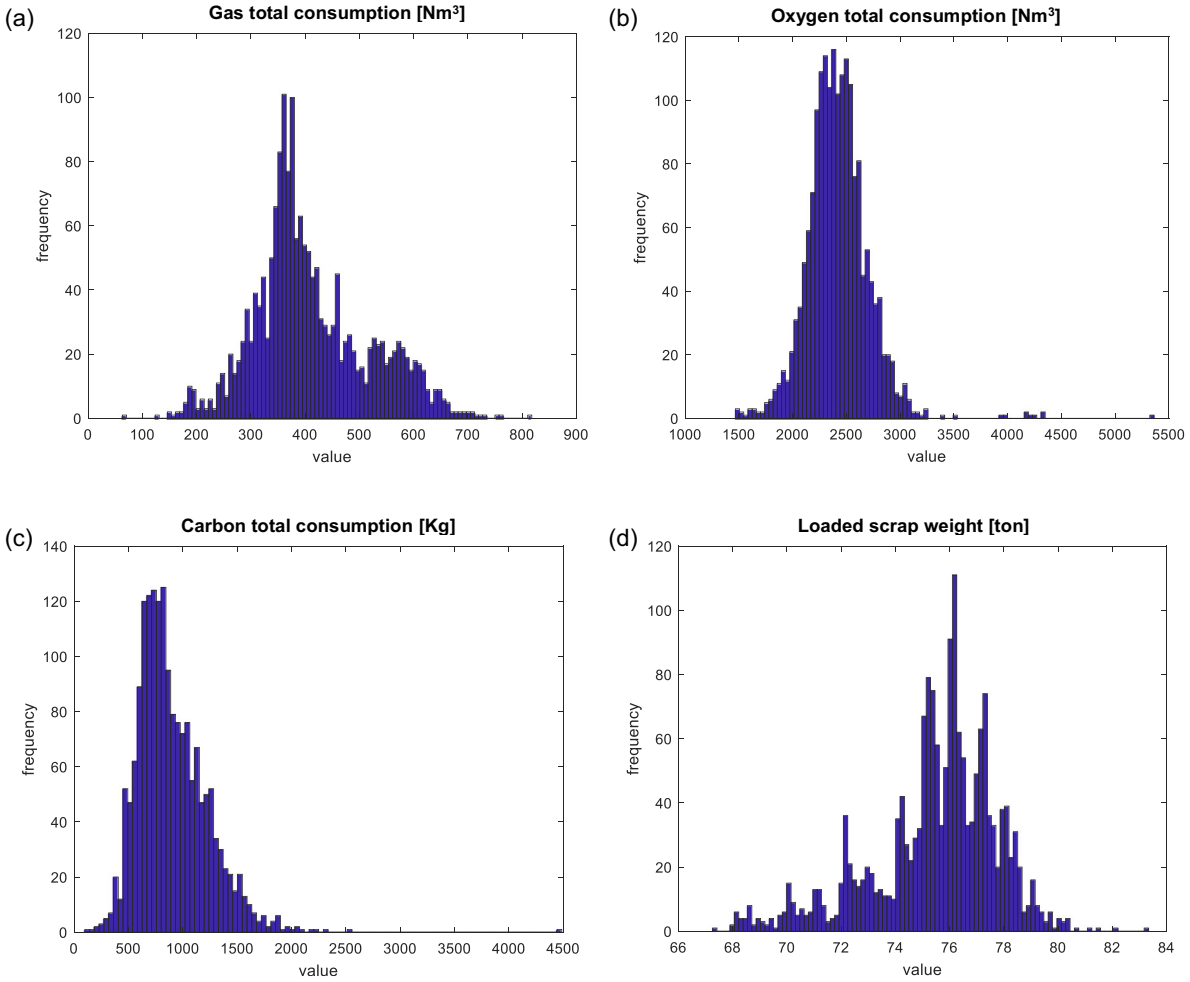

FIGURE 2 | Histograms related to the following parameters of the EAF process: (a) Gas total consumption [Nm^3]; (b) Oxygen total consumption [Nm^3]; (c) Carbon total consumption [Kg]; (d) weight of the loaded scrap [ton].

skewness, tails, and the presence of rare operating regions to confirm that a value can be preliminarily labeled as outliers.

To face this problem, in this work an approach is adopted that combines some of the common outlier detection methods used in the literature and avoids their main limitations by using a fuzzy inference system (FIS) [38]. This approach can be applied to both small and large datasets and follows a 'conservative' strategy, that is, it reduces the chance of missing outliers by jointly considering different definitions of 'outlierness' instead of relying on only one measure. The centroid of the overall data distribution is computed, and a preliminary fuzzy C-Means (FCM) clustering stage [39] is performed on the data. Afterwards, for each sample, 4 normalized features in the range [0,1] are computed: the normalized distance from the overall centroid, the fraction of points in its neighborhood, the normalized mean distance from a subset of the remaining patterns and the degree of membership to the cluster to which it has been assigned by the preliminary FCM. These 4 values are converted into fuzzy linguistic levels (e.g., low/medium/high) and processed by a Mamdani-type FIS. A small set of if-then rules then combines the four features and produces a single outlier index in [0,1], which represents the outlier risk of the pattern. By merging these four complementary indicators through fuzzy rules into a single score, the method provides an interpretable outlier index and reduce the risk of missing potential outliers.

However, outliers are not necessarily unreliable data, as industrial datasets may contain legitimate extreme operating conditions that are infrequent. Therefore, in the present case study

each automatically flagged outlier has been subsequently reviewed by plant operators, to distinguish true process extremes from sensor faults or manual-entry errors, and to decide whether the data can be corrected (sometimes manually entered figures contain trivial errors that can be easily identified) or need to be removed.

Afterwards, to maximize the accuracy of the models while improving their interpretability and numerical stability, irrelevant and redundant variables are eliminated, and the most significant ones are selected. This step is necessary only for the models related to the EAF process, while for the ones related to LF all variables were used.

Redundant variables are identified using the Dominating Set Algorithm (DSA) [40], a method derived from graph theory. According to Pearson's correlation theory, two variables are considered redundant if their linear correlation coefficient ρ is greater than 0.95 and the associated p-value is lower than 0.05. The DSA builds a correlation graph where each feature is represented by a node and an edge links two nodes if they satisfy above stated the redundancy condition ($\rho > 0.95$ and $p < 0.05$). The so-called *minimal dominating set* is the smallest subset of nodes of the graph such that every other node is either selected or directly connected to a selected node. Variables outside the dominating set are considered redundant and removed.

Subsequently, three different methodologies, each belonging to the 3 different categories of filter, wrapper and embedded Variables Selection (VS) methods [41] were used and compared.

Filter VS methods rank predictors independently from the final regressor: here linear correlation was adopted to get a first relevance screening, and RReliefF [42] was used to score the available features based on how well they enable distinguishing sample data showing different values of the variable to predict (more details on Relief are provided in Appendix A). Each method provides an ordered list of variables ranked, respectively, by the absolute linear correlation with the target and by the RReliefF score. The variables that are ranked in the top 10 by at least one of the two methods are included in the filter-based subset.

Wrapper VS methods evaluate candidate input variables using the prediction model itself, treating the model as a black box and keeping variables only if they improve performance on validation data. In the present work, sequential forward selection (SFS) was used [43], which is a greedy step-by-step procedure that starts with an empty set of variables and builds a subset by adding one variable at a time (forward search). At each iteration, all variables not yet selected are tested in turn: one candidate variable is temporarily added to the current subset, the same prediction model is trained on the training data using that subset, and its error is computed on the validation data using the chosen error metric. After trying all remaining variables, the algorithm keeps the variable that yields the lowest validation error. The procedure stops when adding further variables does not improve validation performance, or when another stopping rule is reached (e.g., a maximum number of variables). In the present work, the procedure ends when the improvement in the validation performance is lower than 10^{-6}.

Embedded VS approaches assess the relevance of the input variables directly during model training. In this study, relevance is estimated using a regression decision tree (DT) [44], which highlights important variables because its predictions are built through a sequence of splits of the training data. For each target variable, a regression DT is first trained on the training set. The tree is grown step by step: at each step, the algorithm considers different ways to split the training samples into two groups using one input variable and a cut-off value and then selects the split giving the largest reduction in prediction error at that point in the tree (which means that the target values within the resulting groups become more homogeneous and, thus, easier to predict). This process continues recursively until the tree-growing stopping criteria are met (i.e. maximum possible number of subdivisions the DT reached or the group to be split holding less than 10 samples, or none of the possible subdivision leading to a reduction in prediction error greater than 10^{-6}). After training, the DT is used to generate predictions on the validation set to assess the performance, and the fitted DT provides a measure of variable relevance. Each time an input variable is used to create a split, that split yields a reduction in error; by adding up these error reductions across all splits that use the same variable, an importance score is obtained for each input variable. Variables with higher scores are those that the tree relied on more to reduce prediction error. Finally, variables are selected by retaining those with importance above a predefined threshold (or, alternatively, by selecting the top-ranked variables). In the present work, as threshold value the median of the not null importance scores is used.

The reason to tests an embedded method beside the wrapper one lies in the fact that they differ in how they choose features and in how many variable combinations they try. As a result, the subset showing the best performance during the VS procedure is not necessarily the one with best generalization capabilities on unseen data. In other words, it may happen that even when SFS evaluates feature subsets using the same model family used downstream (e.g., an FFNN), it still selects a subset showing worse generalization. A key reason is selection-induced overfitting: as wrappers repeatedly consult resampling-based estimates (often cross-validation) across many candidate subsets, the search can adapt to noise in those estimates and favor overly optimistic subsets that do not maximize true out-of-sample performance [45, 46]. Moreover, SFS is a nested procedure: once a feature is added it cannot be removed, so the search is algorithmically constrained and may miss better nonnested combinations [47]. By contrast, embedded tree-based methods perform VS during model fitting. DT are classically described as having built-in VS and compared to wrappers can be more data- and computation-efficient, as they avoid retraining a separate predictor for each candidate subset [47]. Finally, differences can also reflect the stability of the VS method: unstable selection under small perturbations of the training set can undermine robustness and reproducibility [48]. Therefore, testing 3 different VS methods helps ensuring that the best subset is finally selected.

2.2 | The Models

Two kinds of ML models were initially preselected, that is, multilayer feed forward NN (FFNNs) [49] and deep echo state Networks (DESN) [50].

FFNNs is the simplest neural architecture and is widely used for solving regression and classification task in the context of supervised learning, that is, where both input and target labeled

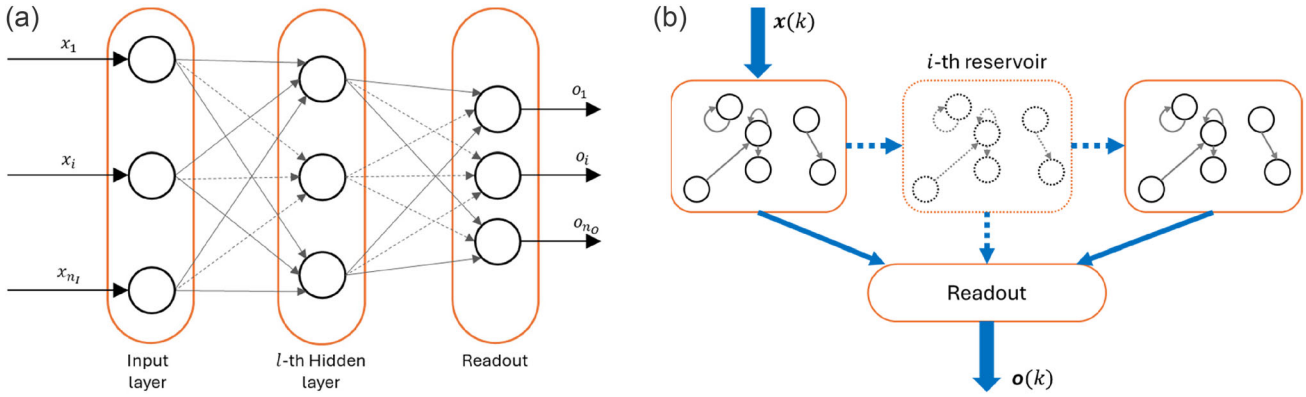

FIGURE 3 | Schematic representation of: (a) a FFNN, (b) a DESN.

data are used for learning nonlinear numerical relationships that maps phenomena (see Figure 3a). In the present work, FFNNs with one hidden layer with standard hyperbolic tangent activation functions are considered, and the model weights are trained through the Levenberg–Marquardt (LM) backpropagation algorithm [49] consistently with the adopted supervised learning setting. A more detailed description of FFNNs and the LM procedure is provided in Appendix B.

DESN [50] are a recurrent neural architecture (RNN) derived from reservoir computing, proposed as evolution of ESN and DL for time series. Similarly to FFNN, the high-level architecture is a sequential network that however, provides memory capacity and so the ability to memorize and predict dynamic systems. A schematic overview of a DESN is shown in Figure 3b, while a detailed description is provided in Appendix C.

Models design implies finding the best architecture for each specific target variable by optimizing the hyperparameters of the model and tuning the model's parameters. The hyperparameters are, for FFNN, the number of neurons in the hidden layer n_h, while for DESN are the number of layers, neurons for each layer, spectral radius, and input scaling factor. For FFNNs and DESNs, the hyperparameters were determined as follows

- The optimal number of hidden neurons of the FFNNs was determined via Grid Search. A set of candidate values of n_h is established by building a nonuniform grid covering the interval [5, 110] with step of 4 neurons in the interval [5, 40] and step of 20 neurons in the interval [50, 110], with a denser grid in the first interval, which had proved to be the optimal range in a preliminary investigation. The search interval is empirically established also on based on the number of data used for the training stage to cover a reasonable model complexity while avoiding overfitting and excessive training time. For each candidate FFNN architecture, 100 independent training procedures are executed, their performance on the validation training set is computed, and the results are aggregated by computing their mean value. The repeated-training and validation-based selection is particularly important to avoid overfitting and to ensure that the selected model complexity remains commensurate with the information content of the industrial data. The architecture (i.e. the number of neuros in the hidden layer) providing the best average performance on validation is selected.
- The hyperparameters of the DESNs were determined via random search by training 1000 different DESN configurations with randomized hyperparameter values and selecting the configuration that provides the best performance.

However, DESNs showed bad performance, as they better fit simulation of dynamical system, for which time series are available. Therefore, the final model implementation focused on FFNNs.

2.3 | The Optimization Approach

The faced optimization problem is synthetically formulated as follows: given an initial production schedule (i.e. a sequence of heats for different steel grades covering a fixed time horizon) covering a time interval $[t_0, t_1]$, finding its optimal arrangement that schedules the more energy intensive grades in the time slots where the cost of energy is lower.

More precisely, we define as 'schedule' over a time interval $[t_0, t_1]$ an ordered sequence of N processing blocks, each consisting of a group of heats characterized by the same steel grade and cast product whose total production time entirely covers the interval $[t_0, t_1]$.

Given an initial schedule S_0, we define its configuration space as the set of all schedules obtained from S_0 by permuting its constituent blocks, which are $N! = N \cdot (N-1) \cdot \ldots \cdot 2 \cdot 1$. In this sense, each block corresponding to a given steel grade is considered 'atomic', as the individual heats within a block are not ordered, as they are produced using the same values of input and process parameters. Since this space is canonically isomorphic to the set of permutations of these blocks, we denote it by $P(S_0)$ (it is worth noting that $S_0 \in P(S_0)$, as it is obtained via the so-called identical permutation). Each element $S \in P(S_0)$ can be associated with a production cost $C(S)$ which strictly depends on the cost of the energy consumed to produce it. To sum up, given an initial schedule S_0 (e.g. manually designed by human operators), the goal of the optimization here is to determine the element of $P(S_0)$ which corresponds to the minimum production cost, namely

$$\min_{S \in P(S_0)} C(S) \tag{1}$$

This is a discrete optimization problem, to be solved by means of a brute-force strategy that exhaustively explores the entire

permutation space, provided that a way is available to compute the cost of a schedule, especially in terms of energy consumption, and to consider the constraints that precisely define the permutation space.

While the total electricity consumption over the time interval $[t_0, t_1]$ is the same for each schedule, the cost might differ if the electricity price varies over time. Therefore, if an even rough forecasting of the time trend of consumption associated to each element of $P(S_0)$ is available, the initial schedule can be arranged at best so that most of the more energy-intensive steel grades are produced at minimal cost. To solve such problem, the simplified and conservative assumption is made that each heat holds a fixed duration and that the power consumption of the EAF and LF processes is constant across such duration, by depending only on the steel grade (and some parameters which refer to the recipe and the process inputs required to produce each steel grade), such as discussed in the previous Subsection 2.1. This leads to overestimating process consumptions, as power off time is not considered, but it is acceptable in the present context because the target is the elaboration of a cost-effective scheduling sequence, and to this aim, a very precise consumption forecast is not needed.

To estimate the total energetic consumption of a schedule, an estimate is needed also for the vacuum degassing (VD) and CC stages. However, as the contribution of these two processes to the total consumption is far lower than the one associated to EAF and LF, and their duration does not significantly change among different steel grades, they are modeled in the simplest possible way, namely as having a constant power consumption and duration (both equal to their average values).

By considering the above stated simplistic assumptions for the electric consumption of the four key processes of the production chain (EAF, LF, VD, and CC), each heat of a given steel grade is associated to 4 constant values for duration and consumptions of these processes. Consequently, each element $S \in P(S_0)$, being a sequence of heats of known steel grades, can be associated to an instantaneous energy consumption function $E_S(t)$, which is piecewise constant, and the total energy cost $C_E(S)$ associated to S can be computed by integrating its instantaneous energy consumption against the electricity price $U(t)$ at time t according to the know tariff plan (which is known and normally is also piecewise constant), as follows

$$C_E(S) = \int_{t_0}^{t_1} E_S(t) \cdot U(t) dt \tag{2}$$

The schedule also needs to be compliant with a series of technical and nontechnical constraints: for instance, a sequence of heats associated with a single steel grade cannot be 'split' in two or more smaller sequences, as this cause inefficiencies at CC. This implies that each block composing the initial schedule S cannot be divided in two parts, i.e. the number of elements of $P(S_0)$ cannot be larger than $N!$ For the same reason, sequences of steel grades associated with different products geometries (e.g. size) cannot be alternated, as this significantly increases the costs of CC configuration and the personnel associated with this task. To this aim, either some elements of $P(S_0)$ are eliminated or huge constant term costs are added to C_E when computing the total cost. Similarly, deadlines for delivery of specific products can represent further constraints, which make some elements of $P(S_0)$ either unfeasible or associated to high additional costs beyond the energy cost. Finally, planned maintenance operations represent a further constraint, as the process needs to be stopped in this period.

To face this complex optimization problem, an ad-hoc optimization algorithm is implemented, which holds a modular structure exploiting basic consolidated elements and has been customized for this specific application. The algorithm, which is schematically depicted in Figure 4, takes as input an initial schedule manually set by operators and consists of four core functions.

- Recognize_blocks.
- Shuffle.
- Permuted_schedule.
- Ranking_and_selection.

The Recognize_blocks function is responsible for analyzing the initial schedule to identify processing blocks, each associated with a specific steel type, that compose it. These blocks are defined as consecutive sequences of operations to produce one steel grade with predetermined process parameters and input feedstocks, and their recognition is essential, as they will be treated as indivisible units during the optimization phase. The function also detects any pauses in processing, distinguishing whether they occur within a single block or between two blocks with different steel grade.

- In the first case, i.e. pauses within a block, they are considered an integral part of the block and are permuted along with it. These interruptions are assumed to be physiological and cannot be managed separately.
- In the second case, that is, pauses between blocks of different types, each interval is treated as an independent block with its own specific duration and can be freely permuted. Each pause is considered as a block and is not split by the optimization.

Additionally, the algorithm allows for optional specification of a custom start and end time, different from the actual schedule. If provided, two supplementary pauses are added outside the blocks, which are also permutable. This feature enables inclusion of operations with personalized start and end times in the optimization process.

The Shuffle function takes as input the blocks identified by the Recognize_blocks function and generates a list of all possible nonrepeating permutations based on them. This list does not directly represent a schedule but serves as blueprint to reconstruct all the possible schedules according to every potential order.

The Permuted_schedule function reconstructs, for each permutation generated by the Shuffle function, a coherent schedule in which operations and pauses are arranged according to the order defined by the permutation itself. The result is a list containing all possible schedules obtained by reordering the initial schedule.

The Ranking_and_selection function ranks the list of admissible schedules produced by the previous function and selects the schedule that is higher in rank. The ranking is based on the

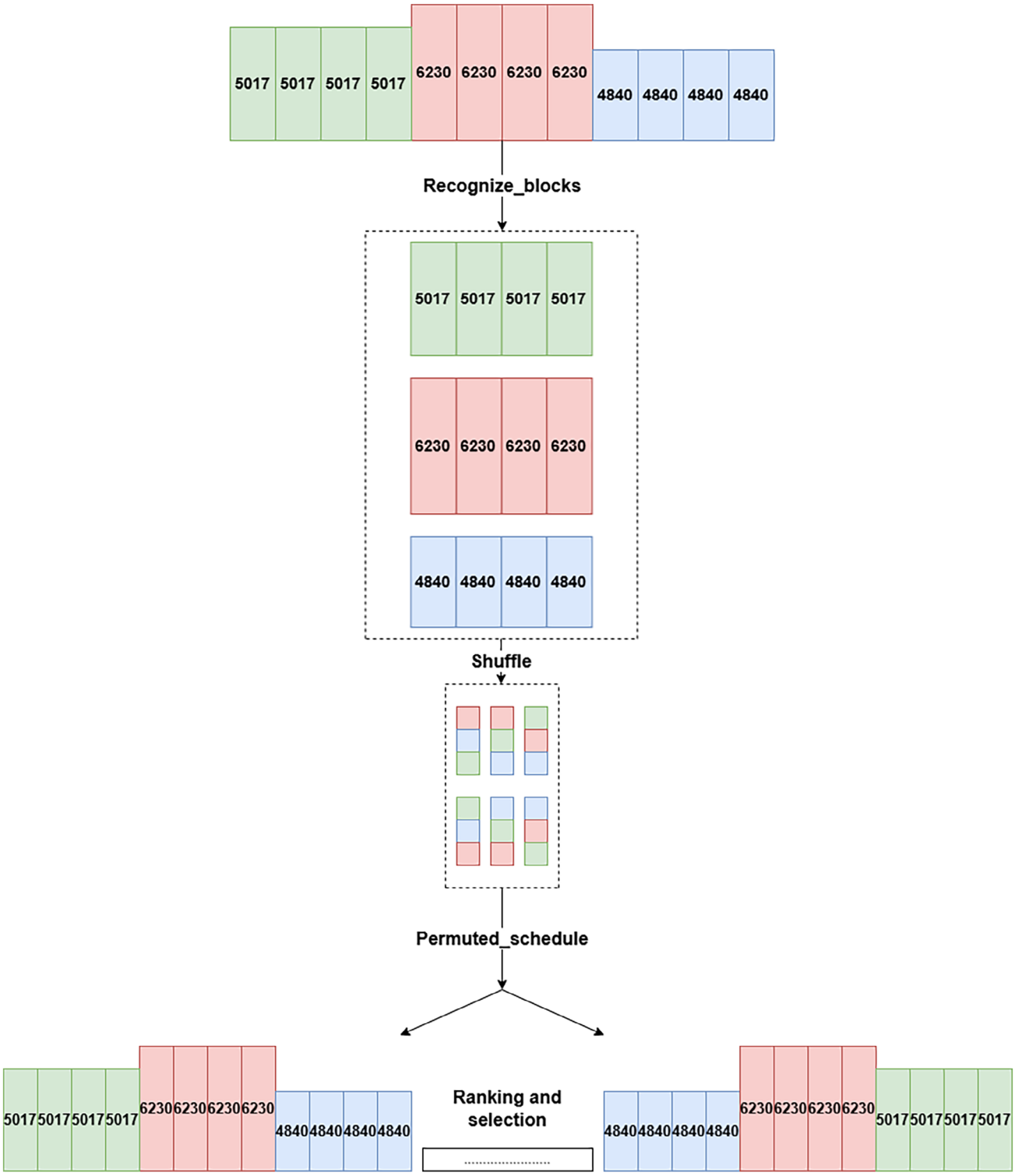

FIGURE 4 | Schematic representation of the optimization approach.

computation of one or more criteria or 'cost functions,' which only depend on the sequence of heats. The cost function is also responsible to account for the practical feasibility of a sequence, by providing unfeasible sequences with a cost value that largely exceeds the mere energy cost (by multiplying it by a factor 10 in the present case) and prevents their selection.

To enhance clarity, pseudocode is presented to illustrate the procedures discussed herein.

RECOGNIZE_BLOCKS(schedule, t_0=None, t_1=None, gap = 15)

‘‘‘

param schedule: the schedule dictionary

:param t_0: start time of the processing

:param t_1: end time of the processing

:param gap: # Pause between heats (15 minutes by default)

:return: list containing the processing blocks

'''

Initialize tol = [] # In this list we store the various types of steel grade and, if present, the pauses between processing block.

Initialize pause_intern = {} # The list stores any pause occurring during a processing block.

for each lavoration:

If SteelGrade changes:

Append processing block to tol #Eventually with its internal pauses

If StartTime != EndTime + gap:

Minutes= EndTime-StartTIme-gap

Append pause to tol

Else

If StartTime != EndTime + gap:

Internal_pause= EndTime-StartTIme-gap

Append Internal_pause to pause_intern

If t_0 is not None and differs from first StartTime:

Initial_pause= StartTime-t_0

Append initial_pause to tol

If t_1 is not None and differs from last EndTime:

Final_pause=t_1-EndTime

Append Final_pause to tol

Return (tol, pause_interne)

SHUFFLE(Steel_type):

'''

Args:

Steel_type (iterable): A list or iterable of elements.

Returns:

list: A list of unique permutations without duplicates.

'''

Convert lists to tuples

Generate all permutations

Remove duplicates using set

Convert back to lists

Return unique permutations

PERMUTED_SCHEDULE(schedule, t_0=None, t_1=None, gap = 15)

'''

:param schedule: the schedule dictionary

:param t_0: start time of the processing

:param t_1: end time of the processing

:param gap: # Pause between heats (15 minutes by default)

:return: A list of permuted schedules.

'''

Initialize perm_list = [] Initializing the list of rearranged schedules

begin_time = t_0 if given ELSE the start time of the first operation in the schedule

(tol, pause_interne) = recognize_blocks(schedule, t_0, t_1, gap)

perm = shuffle(tol)

For each permutation in perm:

new_record= {} #initializing a dictionary for a rearranged schedule

For each block in permutation:

Find block in original Schedule

Aligning the start and end times of the current processing block with the previous in new_record

Append the aligned block to new_record

Append new_record to perm_list

Return perm_list

RANKING_AND_SELECTION(schedule, address_cost_request, t_0=None, t_1=None)

perm_list = permuted_schedule(schedule, t_0, t_1)

cost_dict = cost_function(perm_list, address_cost_request)

cost_list = values(cost_dict)

max_index = argmin(cost_list)

optimized_cost = cost_list[max_index]

best = perm_list[max_index]

Return (best, original_cost, optimized_cost)

As far as the computational complexity is concerned, the brute force approach is clearly cumbersome and might become intractable for large time intervals. However, some practical considerations need to be made for this specific application:

- The permutation is made on blocks of heats and not on single heats, and a sequence of heats associated with a single steel grade cannot be 'split' in two or more smaller sequences, as this cause inefficiencies at CC. Therefore, the number of elements of $P(S_0)$ cannot become larger than $N!$. The duration of 1 heat is normally slightly lower than 1 h (around 56 min in average) and there is a mandatory 'pause time' between two subsequent heats, which here is assumed to be constant (15 min.) This implies that no more than 20 heats/day can be produced even if no interruptions are planned. In practice, blocks are formed by at least 3–4 heats, but much more frequently by a higher number of heats, as frequent changes of production parameters are not convenient from a cost perspective. This implies that, in real conditions, no more than 4–5 blocks are observed over a duration of approximately 24 h.
- The manual schedule S normally covers 1 week, in which at least one 'mandatory' stop is scheduled that cannot be anticipated nor postponed. Therefore, even an optimization covering the whole manual scheduling period can be split at least into two shorter periods, largely limiting the total number of possible schedules to be considered.
- Blocks composed of many heats, (i.e. covering a duration longer than 14 h) are frequent, and it makes no sense to rearrange them over a different time period, as their duration

spans approximately over the whole variation range of hourly energy prices. Therefore, in the optimization procedure, these blocks are kept fixed, while reordering is applied over time intervals which do not include them. In the case of the steelworks considered in the present work, which is representative of a typical EAF-based facility, intervals without large blocks are not longer than 2 days, and more often have a duration lower than 24 h.

- The elaboration of the optimized schedule is not a 'time-critical' task, i.e. computation times even in the order of 10 min are tolerable, as normally the manual elaboration or revision of a production schedule requires more time.
- The operators need to strongly trust the solution provided to accept it, as it replaces the initial choice that was made by them. Complex algorithms are often difficult to interpret for them, while the basic concept of 'considering all the possible solutions' and selecting the best one is easy to understand and does not 'undermine' the confidence in their capabilities and competencies, as the idea that even a skilled human mind cannot consider all possibilities is largely acceptable.

The above listed practical considerations led to privilege, at least for the present application, the brute force approach as it is simple and practical. However, the modularity of the system allows, in future, to replace the brute force algorithm with a more sophisticated approach, for example, based on evolutionary computation or other bio-inspired algorithms, by still exploiting some of its components (e.g. the cost function). Once the solution has gained the trust of the company, they will also be more prone to accept the various 'updates,' once their advantages are clear.

3 | Experimental Results and Discussion

3.1 | Performance of the Models

The data related to 1789 industrial heats were used to design the models: 75% of the available data were used to train the models, while 25% were used in the test phase to assess their performance and select the best model. This sample size is adequate for training compact FFNN regressors, as the model complexity (determined by the number of neurons in the hidden layer n_h) is controlled through the hyperparameter selection and validation. Moreover, the split between training and validation set was performed by considering the distribution of each target and creating empirical clusters by occurrence frequency so that rare operating points were proportionally represented in all subsets and issues due to unbalanced datasets were mitigated.

The following performance indexes were considered

- Mean Absolute Error (MAE), that is, the average value of the absolute error between measured target values and their estimates provided by the model computed on the N samples belonging to test set.
- Maximum Error (MaxErr), that is, the maximum absolute error between measured target values and their estimates computed on the N samples belonging to validation set:

$$\mathrm{MaxErr} = \max_{i=1\ldots N} |y_i - \widehat{y}_i| \tag{3}$$

- Standard Deviation (STD), which provides an indication of how dispersed the estimates provided by the model are.
- Normalized Square Root Mean Square Error (NSRMSE) which is defined as

$$\mathrm{NSRMSE} = \sqrt{\frac{\frac{1}{N}\sum_{i=1}^{N}(\widehat{y}_i - y_i)^2}{\sigma^2}} \tag{4}$$

where σ is the standard deviation of the target variable. As this last index compares the error with the standard deviation of the data, it is the most reliable way to assess the goodness of the model performance.

The preliminary VS stage was able to reduce the number of input variables of 3 out 4 models related to the EAF process, as the best performing models for average power consumption, energy consumption, and heat time hold, respectively, 11, 9 and 7 input variables, while the model for power on time uses all the 19 input variables. Among the 3 tested VS approaches, the embedded one provided the best results. Referring to Table 1, the selected variables were, in decreasing order of importance [2, 1, 9, 18, 5, 22, 11, 17, 8, 3, 12] for average power consumption, [4, 5, 3, 23, 18, 2, 24, 9, 1] for energy consumption, and [5, 4, 3, 18, 2, 7, 14] for heat time.

Given that the target variables are correlated to each other, it is not surprising a subset of variables are, the steel grade (No 1), the total gas and oxygen consumptions of the EAF (Nos. 2 and 3, respectively), the loaded scrap weight (No 5) and the amounts of special demolitions M8 and cast iron-based scrap (Nos. 9 and 18) fed to the EAF are exploited to estimate average power and energy consumptions as well as power on time. VS also confirms that the steel grade and charge basket composition are fundamental determinants of EAF consumption, establishing a direct link between energy predictions and operational practice. This approach naturally captures the metallurgical reality where energy requirements are primarily driven by the choice of steel grade and the corresponding scrap mix strategy. The algorithm highlights the expected correlations between gas/oxygen consumption, total scrap charge, and the energy required to complete the melting process, reflecting the actual operational decisions made at the furnace level. Some scrap mix components show such strict correlation with specific steel grades, being fed exclusively for certain productions, that they become redundant predictors, further validating that the model mirrors real production constraints. Such patterns reflect the actual practice in electric steelworks, where scrap selection is constrained by both technical requirements and economic considerations, and the model successfully learns these implicit decision rules from historical data. In contrast, total process duration shows a more selective correlation pattern, primarily involving charge-related variables such as loaded scrap weight, turnings scrap weight, and cast iron-based scrap weight.

Figures 5 and 6 exemplarily show the performance of some of the developed FFNN-based models as a function of the number of neurons in the hidden layer. The grid search resulted with an optimal number of neurons for each NN architecture. More in details, for EAF models, the optimal number of neurons for

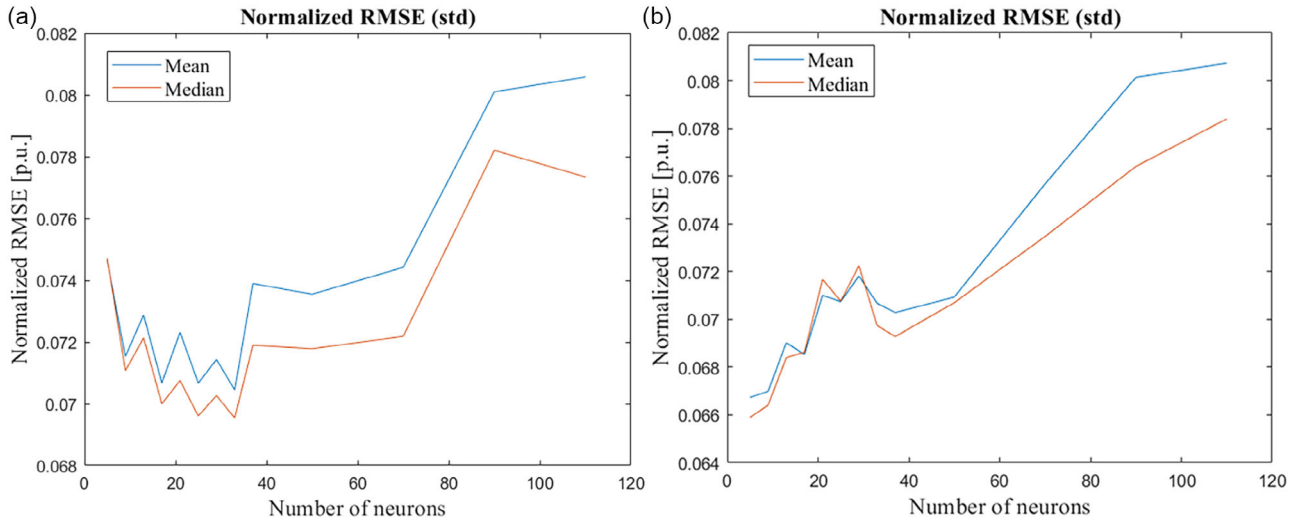

FIGURE 5 | Performance in terms of NSRMSE of the models estimating the average power consumption (a) at the EAF and (b) at the LF.

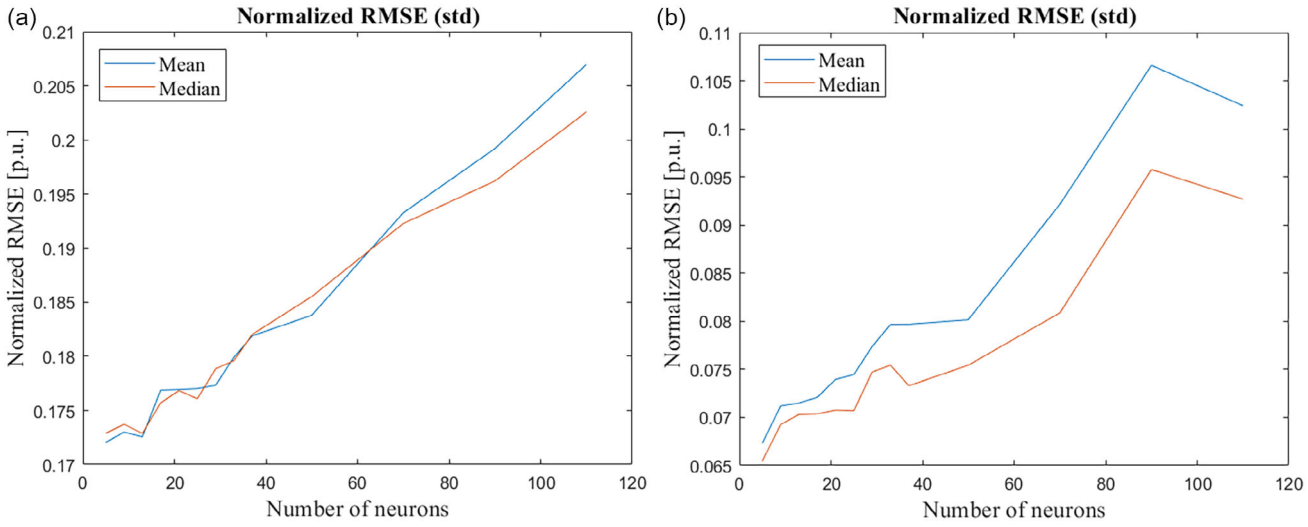

FIGURE 6 | Performance in terms of NSRMSE of the models estimating heat time (a) at the EAF and (b) at the LF.

average power consumption, energy consumption, power-on time and heat time are respectively 32, 13, 10, and 15. For LF models, the optimal number of neurons for average power consumption, energy consumption, power-on time and heat time are respectively 5, 5, 10, and 5.

The results show that increasing the number of neurons beyond a relatively small range does not systematically improve test/validation accuracy in industrial datasets characterized by limited information variability and unbalanced operating conditions. A plausible explanation is that operating conditions are often concentrated around a few stable points, while rare conditions are under-represented. Under these circumstances, larger networks tend to fit noise or unmodelled disturbances (e.g., delays, faults, and nonrepeatable operational events affecting the heat time) rather than improving generalization.

Figure 7 depicts the values of the above listed performance indexes for the selected models related to the EAF and the LF processes (the MaxErr value is multiplied by 0.1 for visualization purposes). The best performance is observed for the models estimating average power consumption and total energy consumption, while the worst performance is observed for the heat time, as this datum is measured in a far less reliable way compared to the other target variables to estimate. Moreover, the inherent complexity of heat time stems from numerous dynamic factors including operator decisions, equipment conditions, and process interruptions that are difficult to capture systematically in operational datasets. Nevertheless, the data-driven approach remains effective for the primary optimization targets, power and energy consumption, demonstrating its practical value even in the presence of incomplete information regarding all process variables.

3.2 | Use of the Models in an Optimization Framework

As a starting point for the optimization, an initial schedule is defined by operators. This leads to introduce the Key Performance Indicator G which is defined based on the total energy cost of the initial schedule $\text{Cost}_{\text{init}}$ and the cost of the

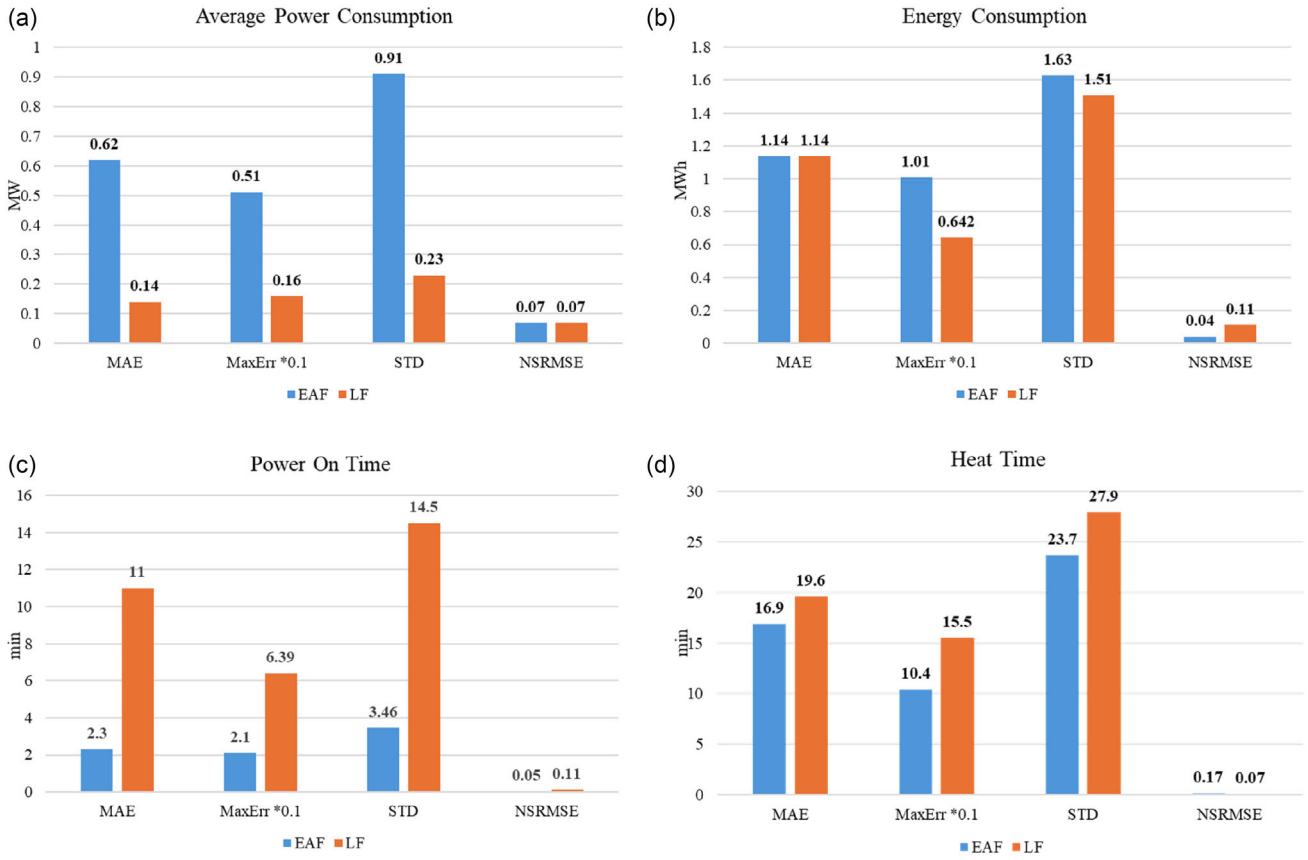

FIGURE 7 | Values of the performance indexes for the selected models estimating: (a) Average Power Consumption; (b) energy Consumption; (c) power On time; and (d) heat Time. NSRMSE is reported on the same diagram for the sake of compactness and completeness, being dimensionless.

reordered sequence $\text{Cost}_{\text{reord}}$, which represents the economic gain $G_{\%}$, and can be defined as

$$G = \frac{\text{Cost}_{\text{record}}}{\text{Cost}_{\text{init}}} \tag{5}$$

In the present work the two costs values reported in Equation (5) coincide with the energy cost C_{E} provided by Equation (2) computed for the initial and reordered schedule, respectively, as neither the operators nor the optimization algorithm selects a schedule which leads, for example, to frequent changes in the CC. However, according to the inspiring principle of modularity and flexibility, the above defined KPI fits well also more complex cost functions. In effects, the computation of costs can include many factors besides the mere energetic costs. For instance, the already mentioned undesirable split of a sequence of heat of the same steel grade or the alternation of steel grades sequences associated with different products geometries (e.g. size), which is here simply prevented by a multiplication factor of the energetic cost, can be accounted in a more exact way via an analytical cost term that really considers the associated personnel and operating costs.

The optimization approach has been validated via on-site acceptance tests (SAT) conducted on an industrial campaign at the electric steelworks, which provided the data for model tuning and test, to assess the capability of the system to support the operators in reducing the energy purchase costs related to a given production lot. The tests were conducted of 4 different

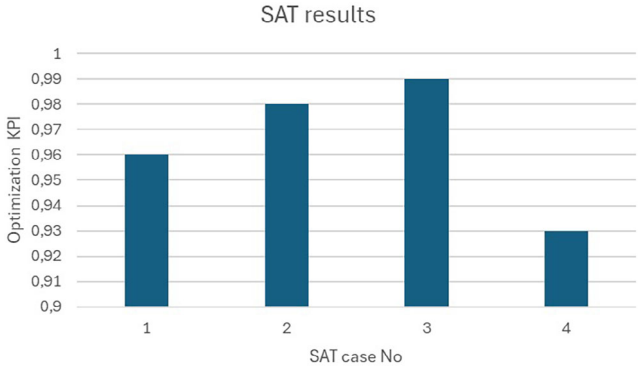

FIGURE 8 | results on the on-SAT of the optimization solution carried out with industrial data.

production campaigns each holding a duration of about 1 day of production. The results are shown in Figure 8.

In all the tests, the schedule proposed by the optimization tool leads to a reduction of electricity costs compared to the original one prepared by the technical personnel, and the electricity cost reduction percentage $\text{ECR}_{\%}$ is connected in a straightforward manner to G as $\text{ECR}_{\%} = 100 \cdot (1-G)$. In particular, $\text{ECR}_{\%}$ reached 7% in the 4$^{\text{th}}$ test, while its minimum value of 1% was observed in the 3$^{\text{rd}}$ test. This high variability of the gain can be due to different and possibly combined factors

- the production mix is not the same for the 4 considered tests: mixes comprising steel grades showing high differences in the associated energy consumptions normally correspond

to higher saving margins. For instance, in test No 3, over 24 h only 3 blocks of heats were present, and the 3 associated steel grades were associated to very similar energy consumption levels. The algorithm is still capable of identifying the best combination by exchanging the order of two blocks compared to the original schedule designed by the operators, but the saving is very limited ($ECR_{\%} = 1\%$).

- The flexibility of the initial schedule S_0 is not the same in the 4 considered tests. For instance, the presence of mandatory production stops can limit the possibility to modify the

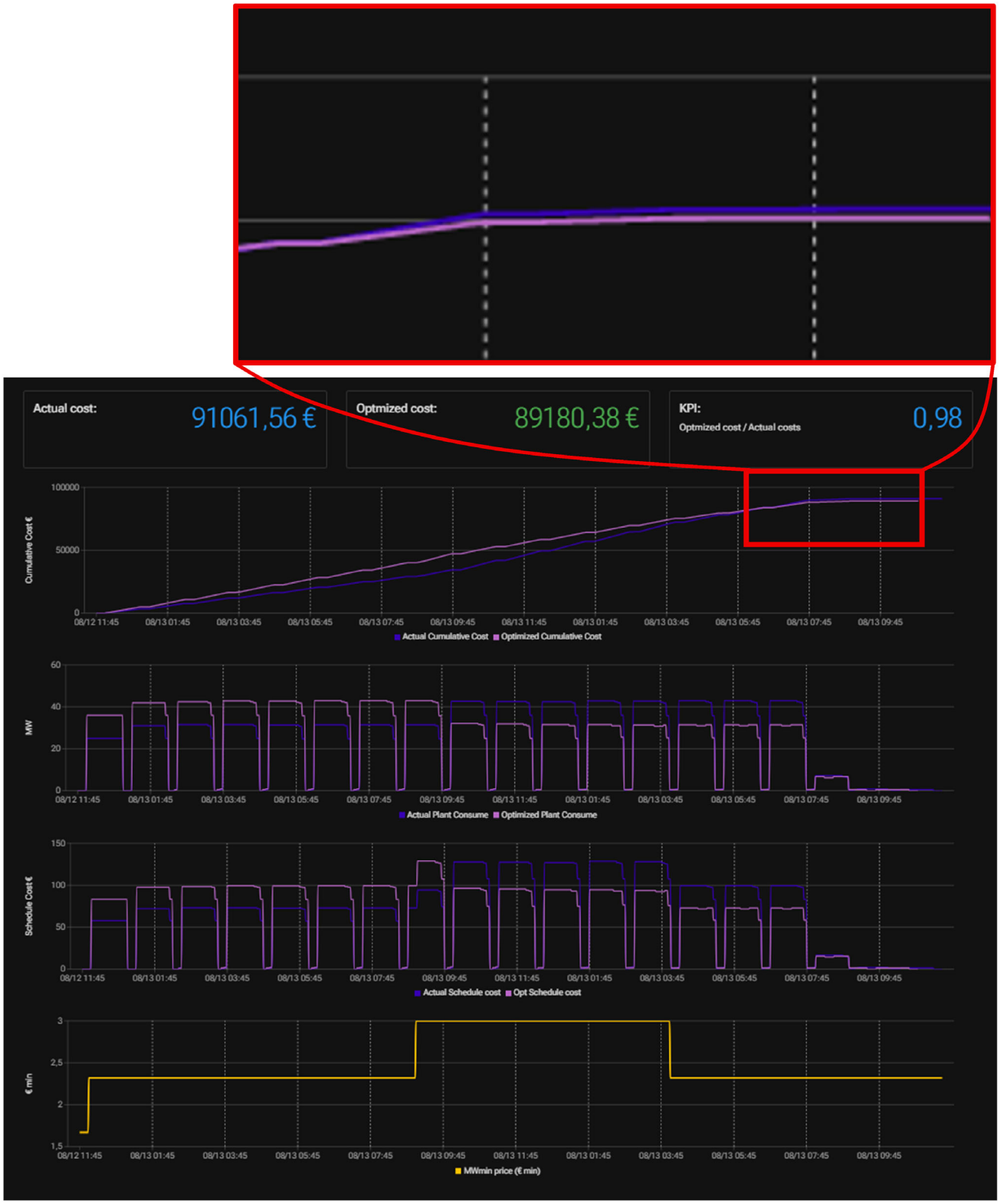

FIGURE 9 | HMI of the developed solution showing a comparison of the estimated consumptions for a non optimized (in violet) and an optimized (in magenta) schedule.

initial schedule, while the presence of flexible downtimes enhances its flexibility. In test No 4, the presence of one flexible downtime of 4 h enables moving some energy-intensive heats in the periods characterized by lower energy costs by obtaining $ECR_{\%} = 7\%$.

- The starting point for the optimization is always an initial schedule, which was S_0 elaborated by human operators. More expert operators are skilled enough to elaborate 'quasi-optimal' schedules, which leave less margin to optimization. Indeed, one of the purpose of digital solutions such as the exemplar one presented here is to ensure stable process performance and mitigate skill gaps, which nowadays is a very relevant issue for the steel sector, such as also attested literature [51–53].

The models and the optimisation tool have been embedded in a software solution built upon an existing commercial software suite that is capable to gather and handle all the data required to evaluate the models thanks to a tailor-made IoT platform [54]. The software is equipped with a human-machine interface (HMI) that should facilitate its exploitation by the technical personnel of the steelworks. Figure 9 shows a window of the developed HMI where the estimated consumptions of about 1 day of production is shown together with the associated cost (the shown cost values are realistic but set for demonstration purposes and do not correspond to actual market numbers, as the company consider these values confidential and cannot share them). The window also enables a comparison between the original schedule that was manually elaborated (violet line) and an optimized schedule rearranged by the embedded optimization algorithm (magenta line). The top diagram of the window depicts the time trend of the cumulative cost with a 'zoom view' (highlighted in red) on the last part of the diagrams: at the end of the considered period the blue curve overcomes the magenta one, because the actual cumulative cost overcomes the optimized one by about 1881€.

The bottom yellow diagram shows the time trend of the energy cost according to the dual rate fee. Figure 9 shows that the sequence of the most energy-intensive heats is shifted in the period of the day where the energy cost is lower, as expected. On top of the diagram, the cost of the actual and optimized schedule is shown, together with the value of the KPI expressed by Equation (5). The fact that the KPI value is lower than one shows that the optimization succeeds in reducing the energy costs, although it does not reduce the consumptions. In this particular case, $G = 0,98$, namely $ECR_{\%} = 2\%$. Therefore, even if the percentage cost savings can appear relatively small, due to the high energy intensity of electric steelmaking process, it translates in a high absolute value. This is even more relevant if one considers that such savings were achieved in a short period of time (less than 1 day) and without any modifications to the plant layout or installation of sensing and monitoring equipment.

The developed solution also comes equipped with other functionalities, for instance, to analyze production and energy data and to appropriately define the energy market.

To sum up, the adopted data-driven methodology prioritizes scalability and practical applicability over theoretical optimization. By using operational data rather than complex physical models of materials and detailed metallurgical principles, the approach enables large-scale industrial deployment while maintaining computational efficiency. This ensures that the optimizer can be readily integrated into existing production workflows without requiring extensive process reengineering or specialized expertise. Importantly, the models demonstrate robust performance even when working with the limited set of variables typically available in industrial environments, making it deployable across different plants without requiring extensive instrumentation upgrades. This characteristic directly supports the development of solutions requiring low capital investments and moderate operator skills while remaining adaptable to various production regimes. The reliance on commonly available process data, such as steel grade, scrap composition, and gas consumption, enhances the transferability of the solution to other electric steelworks, regardless of their specific technological configuration or level of digitalization. The varying influence of different charge mix components on EAF consumption not only confirms metallurgical knowledge but also demonstrates how the model captures operational variability based on component availability and steel grade requirements. As shown by the SAT, this characteristic enables the optimization tool to achieve tangible cost reductions by scheduling energy-intensive heats during periods of lower electricity prices, with the magnitude of savings depending on the flexibility of the production mix and the initial schedule prepared by operators. Future improvements in data acquisition, particularly regarding real-time process events and operator actions, could further enhance the predictive capability for this target variable.

4 | Conclusion

Models for the prediction of electricity consumption in the main processes of the electric steelmaking cycle have been presented that were designed for exploiting NN trained and optimized based on historical process data. The models are fed with data related to the target quality of the steel, the steel recipe in terms of weight and type of used scrap, and the target temperature at each production step. The exploited information is usually available in all electric steelworks, thus the pursued models and, more in general, their design methodology is transferable to other companies. ML-based models are used to minimize the economic impact of production by leveling electricity consumption, as they are part of a more complex decision support tool that helps production managers and operators in optimizing production schedule while complying with production constraints. This solution fits well the demand of a medium size electric steelmaking route and shows enhanced transferability thanks to a modular structure. As it does not require installation of additional hardware, in its current form it is economically viable and particularly suitable to brownfield scenarios. Moreover, in the future the optimization approach can be easily extended to complex production plans characterized by many small size production lots by modifying one of its components, and more complex models, cost, or ranking functions can be included to select the best schedule. In other words, the proposed solution can easily and flexibly follow the evolution and the digitalization of the processes, that can lead to gathering more punctual data and information concerning different aspects of the production process.

Author Contributions

Valentina Colla: conceptualization (equal), data curation (supporting), formal analysis (equal), funding acquisition (lead); investigation (equal), methodology (equal), project administration (lead), resources (lead), software (supporting), supervision (lead), validation (supporting), visualization (supporting), writing – original draft (equal), writing – review and editing (supporting). **Stefano Dettori**: conceptualization (equal), data curation (equal), formal analysis (equal), investigation (equal), methodology (equal), software (lead), validation (equal), visualization (equal), writing – original draft (supporting), writing – review and editing (supporting). **Silvia Cateni**: data curation (lead), formal analysis (equal), investigation (equal), methodology (supporting), validation (supporting), writing – original draft (supporting), writing – review and editing (supporting). **Marco Vannucci**: data curation (equal), investigation (supporting), validation (equal), writing – review and editing (supporting). **Antonella Vignali**: data curation (supporting), software (equal), visualization (equal). **Claudio Mocci**: investigation (equal), software (lead), visualization (equal), writing– review and editing (supporting). **Irene Dovichi**: data curation (equal), software (equal), validation (equal). **Davide Chionna**: conceptualization (equal), methodology (equal), software (equal), writing – original draft (supporting). **Lorenzo Vannini**: data curation (supporting), investigation (equal), validation (lead), writing – original draft (equal). **Enrico Paluzzano**: conceptualization (equal), data curation (equal), funding acquisition (equal), investigation (equal), methodology (equal), project administration (lead), resources (lead), software (lead), supervision (equal), validation (equal), visualization (equal), writing – review and editing (supporting). **Costanzo Pietrosanti**: conceptualization (equal), formal analysis (equal), funding acquisition (lead), methodology (equal), supervision (equal), validation (equal), writing original draft (equal), writing – review and editing (equal). **Davide Onesti**: conceptualization (supporting), software (supporting), visualization (equal). **Davide Venier**: conceptualization (supporting), methodology (supporting), software (equal), validation (supporting).

Acknowledgments

The work described in the present paper has been developed within the project entitled "Energy Management in the Era of Industry 4.0" (Ref. EnerMIND, Grant Agreement No. 899345) that has received funding from the European Union through the Research Fund for Coal and Steel (RFCS), which is gratefully acknowledged. The sole responsibility for the issues treated in the present paper lies with the authors; the Commission is not responsible for any use that may be made of the information contained therein.

Open access publishing facilitated by Scuola Superiore Sant'Anna, as part of the Wiley - CRUI-CARE agreement.

Funding

Research Fund for Coal and Steel of the European Union, Grant Agreement No. 899345.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

Data associated with this article cannot be disclosed due to confidentiality constraints of the involved steelmaking company.

References

1. T. A. Branca, B. Fornai, V. Colla, et al., "Schröder: Industrial Symbiosis and Energy Efficiency in European Process Industries: A Review," *Sustainability* 13, no. 16 (2021): 9159, https://doi.org/10.3390/su13169159.

2. M. Haupt, C. Vadembo, C. Zeltner, and S. Hellweg, "Influence of Input-Scrap Quality on the Environmental Impact of Secondary Steel Production," *Journal of Industrial Ecology* 21, no. 2 (2021): 391, https://doi.org/10.1111/jiec.12439.

3. G. F. Porzio, V. Colla, B. Fornai, M. Vannucci, M. Larsson, and H, "Stripple: Process Integration Analysis and Some Economic-Environmental Implications for an Innovative Environmentally Friendly Recovery and Pre-Treatment of Steel Scrap," *Applied Energy* 161 (2016): 656, https://doi.org/10.1016/j.apenergy.2015.08.086.

4. F. J. Heredia, M. D. Cuadrado Guevara, and C. Corchero, "On Optimal Participation in the Electricity Markets of Wind Power Plants with Battery Energy Storage Systems," *Computers & Operations Research* 96 (2018): 316–329.https://doi.org/10.1016/j.cor.2018.03.004.

5. S. Ashok, "Peak-Load Management in Steel Plants," *Applied Energy* 83, no. 5 (2006): 413, https://doi.org/10.1016/j.apenergy.2005.05.002.

6. Y. Zhang and L. Tang, "Production Scheduling with Power Price Coordination in Steel Industry," in *Asia-Pacific Power and Energy Eng. Conf. APPEEC*, (IEEE, 2010): 5449125, https://doi.org/10.1109/APPEEC.2010.5449125.

7. X. Zhang, G. Hug, and I. Harjunkoski, "Cost-Effective Scheduling of Steel Plants with Flexible Eafs," *IEEE Transactions on Smart Grid* 8 (2016): 1–249, https://doi.org/10.1109/TSG.2016.2575000.

8. A. Haït and C. Artigues, "On Electrical Load Tracking Scheduling for a Steel Plant," *Computers & Chemical Engineering* 35, no. 12 (2011): 3044, https://doi.org/10.1016/j.compchemeng.2011.03.006.

9. K. Nolde and M. Morari, "Electrical Load Tracking Scheduling of a Steel Plant," *Computers & Chemical Engineering* 34, no. 11 (2010): 1899, https://doi.org/10.1016/j.compchemeng.2010.01.011.

10. H. Hadera, R. Labrik, G. Sand, S. Engell, and I. Harjunkoski, "An Improved Energy-Awareness Formulation for General Precedence Continuous-Time Scheduling Models," *Industrial & Engineering Chemistry Research* 55, no. 5 (2016): 1336, https://doi.org10.1021/acs.iecr.5b03239.

11. D. Fraizzoli, D. Ramin, and A. Brusaferri, "A New Modeling Approach to Include EAF Flexibility in the Energy-Aware Scheduling of Steelmaking Process," in *7th Int. Conf. Control Decision & Inf. Tech*, (ELSEVIER, 2020), Vol. 1063, 926398, https://doi.org/10.1109/CoDIT49905.2020.9263981.

12. R. Pan, Z. Li, J. Cao, H. Zhang, and X. Xia, "Electrical Load Tracking Scheduling of Steel Plants under Time-of-use Tariffs," *Computers & Industrial Engineering* 137 (2019): 106049, https://doi.org/10.1016/j.cie.2019.106049.

13. G. Fiorani L. Damiani, R. Revetria, P. Giribone, and M. Schenone, "Models to Estimate Energy Requirements for Iron and Steel Industry: Application Case for Electric Steelworks," *Lecture Notes On Engineering and Computer Science.* 2232 (2017): 920.

14. Z. Li, K. Li, X. Li, C. Huang, and N. Zhang, "Integrating Process-Level Production Scheduling into Bidding Strategy of Steelmaking in Multiple Electricity Markets," *Applied Energy* 402 (2025): 126894, https://doi.org/10.1016/j.apenergy.2025.126894.

15. P. Su, Y. Zhou, and J. Wu, "Multi-Objective Scheduling of a Steelmaking Plant Integrated with Renewable Energy Sources and Energy Storage Systems: Balancing Costs, Emissions and Make-Span," *Journal of Cleaner Production* 428 (2023): 139350, https://doi.org/10.1016/j.jclepro.2023.139350.

16. D. C. Mazur, J. A. Kay, K. D. Mazur, and B. K. Venne, "The Value of Integrating Power and Process for the Metals Industry," *Iron Steel Tech* 15, 5 (2018): 56.

17. R. Kahlid, N. Ahmed, and N. Anglani, "Predictive Model for High Energy Consumption Processes in the Steel Manufacturing Industry Using Machine Learning Tools: Industry 4.0 Pathway," *IEEE Int. Conf.*

on Artificial Intelligence and Green Energy, ICAIGE, (Wiley Online Library, 2024), https://doi.org/10.1109/ICAIGE62696.2024.10776708.

18. U. Camdalı and M. Tunc, "Modelling of Electric Energy Consumption in the AC Electric Arc Furnace," *International Journal of Energy Research* 26 (2002): 935, https://doi.org/10.1002/er.829.

19. V. Logar, D. Dovžan, and I. Škrjanc, "Mathematical Modeling and Experimental Validation of an Electric Arc Furnace," *ISIJ International* 51 (2011): 382, https://doi.org/10.2355/isijinternational.51.382.

20. G. Andonovski and S. Tomažič, Comparison of Data-Based Models for Prediction and Optimization of Energy Consumption in Electric Arc Furnace (EAF)," *IFAC PapersOnLine*, 55. no. 20 (2022): 373, https://doi.org/10.1016/j.ifacol.2022.09.123.

21. P. Kota, A. R. Samiraj, and S. S. Hazra, "Prediction of Electrical Energy Consumption of CONARC Furnace Using Machine Learning Techniques," *International Journal of Simulation and Process Modelling* 19 (2023): 3–203, https://doi.org/10.1504/ijspm.2022.131563.

22. M. Chavosh Nejad, E. Hadavandi, M. M. Nakhostin, and F. Mehmanpazir, "A Data-Driven Model for Energy Consumption Analysis along with Sustainable Production: A Case Study in the Steel Industry," *Energy Sources Part A* 44, no. 2 (2022): 3360, https://doi.org/10.1080/15567036.2022.2064943.

23. C. Chen, Y. Liu, M. Kumar, and J. Qin, "Energy Consumption Modelling Using Deep Learning Technique—A Case Study of EAF," *Procedia CIRP* 72 (2018): 1063, https://doi.org/10.1016/j.procir.2018.03.095.

24. C. Chen, Y. Liu, M. Kumar, J. Qin, and Y. Ren, "Energy Consumption Modelling Using Deep Learning Embedded Semi-Supervised Learning," *Computers & Industrial Engineering* 135 (2019): 757, https://doi.org/10.1016/j.cie.2019.06.052.

25. V. Zawodnik, F. C. Schwaiger, C. Sorger, and T. Kienberger, "Tackling Uncertainty: Forecasting the Energy Consumption and Demand of an Electric Arc Furnace with Limited Knowledge on Process Parameters," *Energies* 17 (2024): 1326, https://doi.org/10.3390/en17061326.

26. H. Lu, H. Zhu, Z. Jiang, H. Li, and C. Yang, "A Novel Hybrid Framework for Precise Electric Energy Consumption Prediction in Steel Production via Electric Arc Furnace: Coupling Mechanistic Models with Advanced Data-Driven Algorithms," *Metallurgical and Materials Transactions B*, 2025, https://doi.org/10.1007/s11663-025-03807-1.

27. L. S. Carlsson, P. B. Samuelsson, and P. G. Jönsson, "Interpretable Machine Learning—Tools to Interpret the Predictions of a Machine Learning Model Predicting the Electrical Energy Consumption of an Electric Arc Furnace," *Steel Research International* 91 (2020): 2000053, https://doi.org/10.1002/srin.202000053.

28. N. Behera, H. Hamed, A. Ghosh, C. Sharma, and SK, "Lahiri: A Hybrid Explainable Machine Learning and Optimization Framework for Energy Reduction in DRI-Based Electric Arc Furnace Steelmaking," *Journal of Sustainable Metallurgy* 11 (2025): 4634, https://doi.org/10.1007/s40831-025-01283-0.

29. P. J. García-Nieto, E. García-Gonzalo, L. A. Menéndez-García, L. Álvarez-de-Prado, M. Menéndez-Fernández, and A. Bernardo-Sánchez, "Interpretable Machine Learning Models for Estimating Electric Energy Consumption in Steel Industries," *Mathematics* 13 (2025): 3364, https://doi.org/10.3390/math13213364.

30. I. Ferretti, F. Zanardi, and L. Zavanella, "Energy Efficiency in a Steel Plant Using Optimization-Simulation," in *Proc. 20th Eur. Modeling and Simulation Symp. EMSS*, (Springer Nature, 2008): 180.

31. X. Li, D. He, Y. Guo, and K. Feng, "Improvement in Stability and Generalization Ability of End-Point Temperature Prediction Model in Ladle Furnace," *Journal of Sustainable Metallurgy* 11 (2025): 4347, https://doi.org/10.1007/s40831-025-01228-7.

32. M. L. Feng, L. Lin, S. He, X.-C. Lin, Z.-X. Hou, and T.-L. Yao, "Temperature Prediction Model for Ladle Furnace Based on Mathematical Mechanisms and the GA–BP Algorithm," *Ironmak. Steelmak* 51, (2024): 692–702, https://doi.org/10.1177/0301923324124024.

33. D. He, C. Song, Y. Guo, and K. Feng, "An Error Correction Method Based on CBR for End Temperature Prediction of Molten Steel in Ladle Furnace," *ISIJ International* 64, no. 8 (2024): 1291, https://doi.org/10.2355/isijinternational.ISIJINT-2024-058.

34. L. Fang, F. Su, Z. Kang, and H. Zhu, "Artificial Neural Network Model for Temperature Prediction and Regulation during Molten Steel Transportation Process," *Processes* 11, no. 6 (2023): 1629, https://doi.org/10.3390/pr11061629.

35. Z.-C. Xin, J.-S. Zhang, J.-G. Zhang, J. Zheng, Y. Jin, and Q. Liu, "Predicting Temperature of Molten Steel in LF-Refining Process Using IF–ZCA–DNN Model," *Metallurgical and Materials Transactions B* 54, no. 3 (2023): 1181, https://doi.org/10.1007/s11663-023-02753-0.

36. K. Parul, A. R. Samiraj, and S. S. Hazra, "Prediction of End Point %C of CONARC Furnace Using Machine Learning Methods," *Sadhana Academy Proceedings in Engineering Science* 48, no. 3 (2023): 132, https://doi.org/10.1007/s12046-023-02163-7.

37. W. Lv and Z. Xie, "Research on End-Point Sulphur Content Prediction Model of Liquid Steel in Ladle Furnace," *Chinese Journal of Scientific Instrument* 35, no. , 6 (2014): 1402.

38. S. Cateni, V. Colla, and M. Vannucci, "A Fuzzy Logic-Based Method for Outliers Detection," *Proc. IASTED Int. Conf. Artificial Intelligence and Applications, AIA*, (IEEE, 2007): vol. 2007, 561.

39. K. Chintalapudi and M. Kam, "Credibilistic Fuzzy c Means Clustering Algorithm," *IEEE Fuzzy Systems Proc* 2 (1998): 2034.

40. J. A. Bondy and U. Murty, *Graph Theory*. (Springer, 2008) ISBN 978-1–84628–969–9.

41. S. Cateni, V. Colla, and M. Vannucci, "A Genetic Algorithm-Based Approach for Selecting Input Variables and Setting Relevant Network Parameters of a SOM-Based Classifier," *International Journal of Simulation: Systems, Science and Technology* 12, no. 2 (2011): 30.

42. M. Robnik-Sikonja and I. Kononenko, "Theoretical and Empirical Analysis of ReliefF and RReliefF," *Machine Learning* 53, no. 1-2 (2003): 23-69, https://doi.org/10.1023/A:1025667309714.

43. D. Ververidis and C. Kotropulos, "Sequential Forward Feature Selection with Low Computational Cost," in *13th IEEE Eur. Signal Proc. Conf*, (Wadsworth International Group, 2005).

44. L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees* (Wadsworth and Books, 1984).

45. J. Reunanen, "Overfitting in Making Comparisons between Variable Selection Methods," *Journal of Machine Learning Research : JMLR* 3 (2003): 1371.

46. J. Loughrey and P. Cunningham, "Overfitting in Wrapper-Based Feature Subset Selection: The Harder You Try the Worse It Gets," *Research and Development in Intelligent Systems XXI. SGAI*, eds. M. Bramer, F. Coenen and T. Allen, 2004, https://doi.org/10.1007/1-84628-102-4_3.

47. I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *Journal of Machine Learning Research : Jmlr* 3 (2003): 1157.

48. S. Nogueira, and G. Brown, "Measuring the Stability of Feature Selection," in *Machine Learning and Knowledge Discovery in Databases. ECML PKDD*, eds. P. Frasconi, N. Landwehr, G. Manco, and J. Vreeken, Lecture Notes in Computer Science 9852 (2016), https://doi.org/10.1007/978-3-319-46227-1_28.

49. H. Yu and B. M. Wilamowski, "Levenberg–marquardt Training," *Intelligent Systems* 12 (2016): 1.

50. S. Dettori, I. Matino, V. Colla, and R. Speets, "A Deep Learning-Based Approach for Forecasting Off-Gas Production and Consumption in the Blast Furnace," *Neural Computing & Applications* 34, no. 2 (2022): 911, https://doi.org/10.1007/s00521-021-05984-x.

51. H. Morioka, K. Sakakibara, M. Nakamura, and S. Yamada, "Development of a Talent Management Support System for Sustaining Organizational Human Capability," in *34th Int. Conf .Flexible Autom. Intell. Manuf. FAIM*, Lecture Notes in Mechanical Engineering, (MDPI, 2025): 614, https://doi.org/10.1007/978-3-032-07675-5_61.

52. K. Maldonado-Mariscal, M. Cuypers, A. Götting, and M. Kohlgrüber, "Skills Intelligence in the Steel Sector," *Machines* 11 (2023): 335, https://doi.org/10.3390/machines11030335.

53. T. A. Branca, B. Fornai, V. Colla, et al., "Skills Demand in Energy Intensive Industries Targeting Industrial Symbiosis and Energy Efficiency," *Sustainability* 14, no. 23 (2022): 15615, https://doi.org/10.3390/su142315615.

54. C. Pietrosanti, E. Paluzzano, G. Planu, M. Ometto, V. Colla, and A. F. N. Djangang, "A Metal Tailor-Made IoT Platform to Enable Power Management Integration and Support Procurement Access to the Power Market," in *AISTech - Iron and Steel Tech. Conf. Proc.*, (Springer, 2025): 2344, https://doi.org/10.33313/389/249.

55. C. Gallicchio, A. Micheli, and L. Pedrelli, "Echo State Property of Deep Reservoir Computing Networks," *Cognitive Computation* 9, no. 3 (2017): 337, https://doi.org/10.1007/s12559-017-9461-9.

Appendix A

Relief-based features ranking

Relief-based features ranking methods are a category of filter approaches that assigns a relevance score to each input variable before training any predictive model for regression or classification tasks. In particular, for classification the most widely adopted method of the family is *ReliefF*, while for regression *RReliefF* is mostly widespread. Their output is typically a ranking of the potential variables used to select a subset of most influential ones.

The basic principle is the same for whole family of algorithms: let us consider a modelling problem where a target variable y (categorical in the case of classification, continuous in the case of regression) must be estimated using a set of N_F variables or features $\boldsymbol{X} = (x_1, \ldots x_{N_F}) \in \mathbb{R}^{N_F}$. To design any kind of supervised ML-based model, a training set composed of N_T couples $(\boldsymbol{X_1}, y_1) \ldots (\boldsymbol{X}_{N_T}, y_{N_T})$ is available. Roughly speaking, a feature is considered 'influential' if it shows different values for couples characterised by different target values, and similar values for couples characterised by similar or same target values. This 'local' viewpoint is often more informative than purely global criteria (such as linear correlation), because it can highlight variables that matter in specific regions of the feature space and can capture nonlinear effects. The algorithm returns a vector $\boldsymbol{S} = (s_1, \ldots s_{N_F})$ of so-called 'scores' associated to each feature that represents its significance for determining the target and are computed through an iterative procedure, which involves a reference dataset $\mathcal{D}$ composed of $M \leq N_T$ samples and is articulated into M steps. Initially, all scores are null. At the generic step r, the algorithm takes one sample $(\boldsymbol{X}_r, y_r) \in \mathcal{D}$ and finds the K nearest neighbours of $\boldsymbol{X}_r$ in the feature space, where K is a user-defined integer.

In classification, *ReliefF* at the r-th step updates for K times the score s_j of the j-th feature by checking whether the class y_k associated to each neighbor $\boldsymbol{X}_k (k = 1 \ldots K)$ coincides or not to the class y_r of the reference sample, according to the following equation:

$$s_j^{(r,k)} = \begin{cases} s_j^{(r,k-1)} - \frac{\Delta_j(x_{j,r}, x_{j,k})}{M} w_{rk} & \text{if } y_k = y_r \\ s_j^{(r,k-1)} + \frac{\pi(y_k)}{1-\pi(y_r)} \frac{\Delta_j(x_{j,r}, x_{j,k})}{M} w_{rk} & \text{otherwise} \end{cases} \quad \text{for } k = 1 \ldots K \tag{A1}$$

where $\Delta_j(x_{j,r}, x_{j,k})$ quantifies the difference in feature x_j between the reference $\boldsymbol{X}_r$ and the neighbor $\boldsymbol{X}_k$, w_{rk} is a weight controlling the influence of the neighbor, while $\pi(y_k)$ and $\pi(y_r)$ are the values of the prior probability of occurrence of the classes y_k and y_r, which are typically estimated from their presentation frequencies in the dataset and are included to reduce bias when class proportions are not balanced. In deeper detail, the term $\Delta_j(x_{j,r}, x_{j,k})$ is defined according to the type of feature x_j as follows:

$$\Delta_j(x_{j,r}, x_{j,k}) = \begin{cases} 0 & \text{if } x_{j,r} = x_{j,k} \\ 1 & \text{if } x_{j,r} \neq x_{j,k} \end{cases} \quad \text{if } x_j \text{ is discrete} \tag{A2}$$

$$\Delta_j(x_{j,r}, x_{j,k}) = \frac{|x_{j,r} - x_{j,k}|}{\max_{\mathcal{D}} x_j - \min_{\mathcal{D}} x_j} \text{ if } x_j \text{ is continuous} \tag{A3}$$

The normalization in Equation (A3) makes $\Delta_j(x_{j,r}, x_{j,k})$ dimensionless and comparable across variables in different scales. The weight w_{rk}, depends on the rank of $\boldsymbol{X}_k$ within the list of the K nearest neighbours of $\boldsymbol{X}_r \text{rank}(r,k) \in \{1, \ldots, K\}$ (being 1 associated to the closest neighbor and k to the farthest), and is normalized so that the sum of the neighbor weights is unitary for each reference sample.

$$w_{rk} = \frac{e^{-\left(\frac{\text{rank}(r,k)}{\sigma}\right)^2}}{\sum_{h=1}^{K} e^{-\left(\frac{\text{rank}(r,h)}{\sigma}\right)^2}} \tag{A4}$$

The parameter $\sigma > 0$ controls how quickly influence decreases with rank: large values of σ make w_{rk} close to the uniform value $1/K$, while small σ values emphasize the closest neighbours.

Analogously, for regression tasks, at the r-th step *RReliefF* updates for K times 3 intermediate accumulators $A_j^{(r,k)}$, $B_j^{(r,k)}$, and $C_j^{(r,k)}$ based on the r-th reference input vector $\boldsymbol{X}_r$ and each of its K neighbours $\boldsymbol{X}_k (k = 1 \ldots K)$ as follows:

$$\begin{aligned} A^{(r,k)} &= A^{(r,k-1)} + \Delta(y_r, y_k) w_{rk} \\ B_j^{(r,k)} &= B_j^{(r,k-1)} + \Delta_j(x_{j,r}, x_{j,k}) w_{rk} \text{ for } k = 1 \ldots K \\ C_j^{(r,k)} &= C_j^{(r,k-1)} + \Delta(y_r, y_k) \Delta_j(x_{j,r}, x_{j,k}) w_{rk} \end{aligned} \tag{A5}$$

where w_{rk} is defined in Equation (A4), is defined in Equations (A2) and (A3), while $\Delta(y_r, y_k)$ measures the difference in the target values y_r and y_k that are associated, respectively, with the reference input vector $\boldsymbol{X}_r$ and its k-th neighbor $\boldsymbol{X}_k$, and is computed as

$$\Delta(y_r, y_k) = \frac{|y_r - y_k|}{\max_{\mathcal{D}} y - \min_{\mathcal{D}} y} \tag{A6}$$

After all updates have been performed over the M reference samples, the final score for each feature j is computed using the final values A_j, B_j and C_j of the accumulators in Equation (A5) as follows

$$s_j = \frac{C_j}{A} - \frac{B_j - C_j}{M - A}. \tag{A7}$$

In the present work, which deals with a regression task, *RReliefF* was adopted: the reference dataset $\mathcal{D}$ is composed of 80% of the training data, which are 75% of the all the data available, therefore $N_{/} = 1340$, $M = 1072$ and the internal parameters of algorithms were set to $K = 20$ and $\sigma = 50$.

Appendix B

Feed Forward Neural Networks

FFNNs are characterized by two features: (i) the network is organized in layers, and (ii) the input information is propagated in one single direction, from the input layer to the readout (the last one). These characteristics provide several advantages during the learning process as each layer learns in a hierarchical fashion, recognizing increasingly complex and abstract representation of the input data, starting from local simple

patterns to higher complex representations, sequentially extracted along the data flow from input to the readout. The layers are connected in series: the input receives the input raw data, the hidden layers extract hierarchical features from data flow, the readout combines the last feature representation for calculating the outputs. A graphical representation of FFNN is shown in Figure 3a, highlighting the sequential computational graph from the input vector $\boldsymbol{x}$ to the output vector $\boldsymbol{o}$. The mathematical representation is composed of a set of equations. For the first hidden layer the output $\boldsymbol{o}^{(1)}$ is computed as follows

$$\boldsymbol{o}^{(1)} = \phi^{(1)}\left(\boldsymbol{z}^{(1)}\right) \tag{B1}$$

$$\boldsymbol{z}^{(1)} = \boldsymbol{W}^{(1)}\boldsymbol{x}^{(1)} + \boldsymbol{b}^{(1)} \tag{B2}$$

where $\phi^{(i)}$ is the activation function of the i-th layer (typically ReLU and hyperbolic tangent tanh), $\boldsymbol{z}^{(i)}$ is the activation vector, $\boldsymbol{x}^{(1)}$ is the input vector composed of n_{I} components, $\boldsymbol{W}^{(i)}$ and $\boldsymbol{b}^{(i)}$ are respectively the weight matrix and the bias vector, which are the target of the training procedures. The output of the following hidden layers is computed as

$$\boldsymbol{o}^{(l)} = \phi^{(l)}\left(\boldsymbol{z}^{(l)}\right) \tag{B3}$$

$$\boldsymbol{z}^{(l)} = \boldsymbol{W}^{(l)}\boldsymbol{o}^{(l-1)} + \boldsymbol{b}^{(l)} \tag{B4}$$

The readout output $\boldsymbol{y}$ of a network composed of n_{h} hidden layers is calculated as follows

$$\boldsymbol{y} = \phi^{\mathrm{R}}\left(\boldsymbol{o}^{(n_{\mathrm{h}})}\right) \tag{B5}$$

For regression tasks, ϕ^{R} is typically linear (the identity), while for classification, softmax or sigmoid are the common ones.

The training procedure of a FFNN is typically carried out through supervised learning methodologies where, given a dataset of tuples of data composed of inputs and correlated labelled targets, the learning algorithm minimizes an objective function (or loss function) to correctly generalize to new data. The state of the art for solving classification and regression tasks is based on the back propagation, an algorithm that calculates in an efficient way the gradient of the loss function with respect to the parameters, then updated through gradient descent-based methods. Several efficient algorithms exist, and the selection fully depends on the dimension of the dataset, the smoothness of the loss function with respect to the learnable parameters, and the number of parameters to adapt. Many physical and industrial modelling/regression tasks can be formulated as non-linear least square problems, as the hidden physical phenomena often exhibits continuous relatively smooth relationships between inputs and outputs, for which the Jacobian is typically well defined and not noisy. In this context, Levenberg–Marquardt (LM) algorithm [49] is an efficient choice when the number of samples of the training dataset and the number of free parameters of the network is low (i.e. low number of neurons and hidden layers). LM is a variant of the Gauss-Newton algorithm where the hessian is approximated, deriving the following update step for calculating the update of the parameters

$$\Delta\boldsymbol{\theta}_k = -\left[\boldsymbol{J}^{\mathrm{T}}(\boldsymbol{\theta}_k)\boldsymbol{J}(\boldsymbol{\theta}_k) + \mu_k\boldsymbol{I}\right]^{-1}\boldsymbol{J}(\boldsymbol{\theta}_k)\boldsymbol{e} \tag{B6}$$

where $\boldsymbol{e}$ is the regression error, $\boldsymbol{J}(\boldsymbol{\theta}_k)$ is the Jacobian of the errors with respect to the learnable parameters $\boldsymbol{\theta}_k$, and μ_k is an adaptive parameter of LM used for regularizing the inversion of the $\boldsymbol{J}^{\mathrm{T}}(\boldsymbol{\theta}_k)\boldsymbol{J}(\boldsymbol{\theta}_k) + \mu_k\boldsymbol{I}$ matrix. LM algorithm is a robust algorithm for training FFNNs, as it balances qualities of the gradient descent methods and the convergence speed of Gauss-Newton, but at the cost of strong memory requirements, resulting efficient only for relatively small-medium size datasets and small networks, such as the ones considered in the present work.

Appendix C

Deep Echo State Networks

Similarly to FFNN, DESN are a sequential stack of layers called reservoirs, where each internal state is updated according to a discrete dynamic system. For the first reservoir

$$\boldsymbol{h}^{(1)}(k) = \phi^{(1)}\left(c_{\mathrm{in}}^{(1)}\boldsymbol{W}_{\mathrm{in}}^{(1)}\boldsymbol{x}(k) + \boldsymbol{W}_{\mathrm{r}}^{(1)}\boldsymbol{h}^{(1)}(k-1)\right) \tag{C1}$$

where $\boldsymbol{h}^{(1)}(k)$ is the state vector at the k-th time step, $\boldsymbol{x}$ is the input vector, $\phi^{(i)}$ is the activation function of the i-th reservoir, and $c_{\mathrm{in}}^{(i)}$, $\boldsymbol{W}_{\mathrm{in}}^{(i)}$ and $\boldsymbol{W}_{\mathrm{r}}^{(i)}$ are respectively the input scaling factor, the input weighting matrix and reservoir matrix of the i-th reservoir. For the following reservoirs, the state of the i-th reservoir is updated according to the following equation

$$\boldsymbol{h}^{(i)}(k) = \phi^{(i)}\left(c_{\mathrm{in}}^{(i)}\boldsymbol{W}_{\mathrm{in}}^{(i)}\boldsymbol{h}^{(i-1)}(k) + \boldsymbol{W}_{\mathrm{r}}^{(i)}\boldsymbol{h}^{(i)}(k-1)\right) \tag{C2}$$

where the input of the layer is the state of the previous $i-1$-th reservoir.

Finally, the readout of the network combines the state vector of each reservoir for calculating the output $\boldsymbol{o}(k)$ for the k-th time step. For a DESN with N reservoirs connected in series

$$\boldsymbol{h}(k) = \left[\boldsymbol{h}^{(1)}(k)^{\mathrm{T}} \quad \dots \quad \boldsymbol{h}^{(N)}(k)\right]^{\mathrm{T}} \tag{C3}$$

$$\boldsymbol{o}(k) = \phi^{O}(\boldsymbol{W}^{O}\boldsymbol{h}(k)) \tag{C4}$$

where ϕ^{O} is the activation function of the readout and $\boldsymbol{W}^{O}$ is the weighting matrix of the readout. For time series forecasting, ϕ^{O} is typically linear (identity).

The training of DESN is quite straightforward. In the first phase, all the weights of the reservoirs are randomly initialized, with the only constraint of verifying the Echo State Property (ESP) [55].

For DESN, a necessary condition for verifying the ESP is the following

$$\rho\left(\boldsymbol{W}_{\mathrm{r}}^{(i)}\right) \leq 1 \tag{C5}$$

where $\rho(\boldsymbol{W}_{\mathrm{r}}^{(i)})$ is the spectral radius of the reservoir matrix $\boldsymbol{W}_{\mathrm{r}}^{(i)}$. For achieving this, each reservoir matrix is set starting from a randomized sparse $\widetilde{\boldsymbol{W}}_{\mathrm{r}}^{(i)}$ scaled for its spectral radius and multiplied for a desired spectral radius $\widetilde{\rho}^{(i)} \leq 1$

$$\boldsymbol{W}_{\mathrm{r}}^{(i)} = \widetilde{\rho}^{(i)}\frac{\widetilde{\boldsymbol{W}}_{\mathrm{r}}^{(i)}}{\rho\left(\widetilde{\boldsymbol{W}}_{\mathrm{r}}^{(i)}\right)} \tag{C6}$$

ESP allows creating stable dynamics withing the reservoir that can be combined by the readout.

Once the network is initialized, $\boldsymbol{W}^{O}$ can be calculated through ridge regression algorithm

$$\boldsymbol{W}^{O} = \overline{\boldsymbol{Y}}\boldsymbol{H}^{\mathrm{T}}(\boldsymbol{H}\boldsymbol{H}^{\mathrm{T}} + \lambda\boldsymbol{I})^{-1} \tag{C7}$$

where $\overline{\boldsymbol{Y}}$ and $\boldsymbol{H}$ are, respectively, the target time series and the state time series of $\boldsymbol{h}$ calculated along the input time series. λ is the regularization coefficient of the ridge regression algorithm. Common DESN training algorithms does not necessarily need back propagation for calculating the model parameters, avoiding numerical problems typically encountered while using standard RNNs, providing users with a simple but effective modelling tool for time series and dynamical systems.