Facultad de Ciencias

Inteligencia artificial para optimizar el consumo de energía en el proceso de laminación del acero

(Artificial Intelligence to Optimize Energy Consumption in the Steel Rolling Process)

Trabajo de Fin de Máster
para acceder al

MÁSTER EN DATA SCIENCE

Autor: Sergio Bolívar Gómez

Directora: Lara Lloret Iglesias

Codirector: Diego Ferreño Blanco

Julio 2023

Agradecimientos

Quisiera expresar todo mi agradecimiento a mi directora, Lara. Muchas gracias por haberme dado la oportunidad de trabajar en este proyecto, por confiar en mí desde el primer momento, y por tu inestimable ayuda siempre que la he necesitado. Muchas gracias también por tu gran esfuerzo e implicación en la organización del máster, siempre dispuesta a echarnos una mano. Ha sido un gusto trabajar contigo.

Asimismo, me gustaría agradecer a Diego, mi codirector, su labor como supervisor y su papel como intermediario entre las distintas partes involucradas en el proyecto. También me gustaría dar las gracias a Jose por su ayuda en el modelado de elementos finitos y sus comentarios e ideas para mejorar el trabajo. Gracias a Global Steel Wire, especialmente a Estela, por haber proporcionado los datos.

Muchas gracias a mis compañeros del máster, ha sido un placer conoceros. Estoy seguro que esta etapa no hubiese sido lo mismo sin vosotros. ¡Mucha suerte a todos!

Por último, pero no por ello menos importante, muchas gracias a mi familia y amigos por haberme acompañado durante todos estos años. Gracias a vuestra ayuda y compañía todo ha sido un poco más fácil.

A todos, muchísimas gracias.

Abstract

Wire rod steel rolling is a highly complex metallurgical process involving numerous phases. One of the most critical is the heating phase, where steel billets (semi-finished steel products) undergo various heating treatments in an induction furnace. These heating treatments enable their subsequent shaping in the rolling phase, and determine the quality of the final product.

This research focuses on introducing a novel methodology that integrates finite element modeling, machine learning, and reinforcement learning to optimize energy consumption during the heating phase of the wire rod steel rolling process. Specifically, the proposed methodology consists of three major phases. In the first phase, a finite element digital twin of an induction furnace has been developed that accurately simulates the heating process of the billets. In the second phase, two machine learning models based on convolutional and/or recurrent neural networks have been implemented. These models can predict the temperature profiles of the billets at the induction furnace's exit with satisfactory precision. Finally, in the third phase, a reinforcement learning framework based on deep Q-learning has been proposed to identify heating patterns that reduce the energy consumption of the furnace.

These goals have been accomplished using real-world data provided by Global Steel Wire, a steelmaker located in Santander (Cantabria, Spain). This data mainly comprises multivariate time series related to the instantaneous power of the furnace's inductors and the advance of the billets in the furnace.

Keywords: energy consumption optimization; artificial intelligence; reinforcement learning; deep q-network; deep q-learning; machine learning; recurrent neural networks; LSTM; convolutional neural networks; Industry 4.0; steel rolling; induction furnace.

Resumen

El laminado de acero en alambrón es un procedimiento siderúrgico altamente complejo en el que están involucradas numerosas fases. Una de las más críticas es la fase de calentamiento, donde las palanquillas de acero (productos semielaborados de acero) se someten a diversos tratamientos térmicos en un horno de inducción que permiten su posterior moldeado y determinan la calidad del producto final.

Este trabajo tiene como objetivo introducir una metodología novedosa que integra el modelado por elementos finitos, el aprendizaje automático y el aprendizaje de refuerzo para optimizar el consumo de energía durante la fase de calentamiento del proceso de laminado en alambrón. En particular, la metodología propuesta consta de tres fases principales. En la primera fase, se ha desarrollado un gemelo digital por elementos finitos de un horno de inducción para simular con precisión el proceso de calentamiento de las palanquillas. En la segunda fase, se han implementado dos modelos de aprendizaje automático basados en redes neuronales convolucionales y/o recurrentes que son capaces de predecir los perfiles de temperatura de las palanquillas a la salida del horno con una precisión satisfactoria. Finalmente, en la tercera fase, se ha propuesto un marco de trabajo de aprendizaje por refuerzo basado en deep Q-learning para identificar patrones de calentamiento de las palanquillas que reduzcan el consumo energético del horno.

Estos objetivos se han logrado utilizando datos reales proporcionados por Global Steel Wire, una empresa siderúrgica situada en Santander (Cantabria, España). Estos datos consisten principalmente en series temporales multivariantes relacionadas con la potencia instantánea de los inductores del horno y el avance de las palanquillas en el horno.

Palabras clave: optimización del consumo de energía; inteligencia artificial; aprendizaje por refuerzo; deep q-network; deep q-learning; aprendizaje automático; redes neuronales recurrentes; LSTM; redes neuronales convolucionales; Industria 4.0; laminado de acero; horno de inducción.

Contents

Acronyms

The following acronyms have been used throughout this master's thesis:

AI	Artificial Intelligence
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
CV	Computer Vision
DNN	Deep Neural Network
DQN	Deep Q-Network
DQL	Deep Q-Learning
EU	European Union
FE	Finite Element
FCNN	Fully Connected Neural Network
GDP	Gross Domestic Product
GPU	Graphics Processing Unit
ICT	Information and Communication Technologies
IoT	Internet of Things
LSTM	Long Short-Term Memory
MAPDL	Mechanical ANSYS Parametric Design Language
ML	Machine Learning
MSE	Mean Squared Error
QL	Q-Learning
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
GSW	Global Steel Wire

Chapter 1

Introduction

The industrial sector plays a key role in the global economy, being the second largest contributor to the Gross Domestic Product (GDP) of the European Union (EU) after the services sector [1]. It encompasses a wide range of industries, such as textiles, food processing, petrochemicals, and metallurgy, among others. However, it is also one of the largest energy consumers within the EU, accounting for 25.6% of the final energy consumption in 2021 [2]. To address this, several green policies have been implemented, including the 2030 Agenda [3], the Paris Agreement [4], and the European Green Deal [5], which aim to promote sustainable and energy-efficient industrial practices to reduce energy consumption, thereby curbing climate change.

Recently, the industrial sector has undergone a transition towards the so-called Industry 4.0 [6], mainly driven by the implementation of green policies such as those mentioned earlier. This transition has also been accelerated by the recent war in Ukraine, which has caused global energy strategies to be rethought as many manufacturers have been forced to reduce their production and even close down due to rising energy prices and resource scarcity. This transformation process towards Industry 4.0 entails greater integration of information and communication technologies (ICT) into production systems, seeking greater energy efficiency and productivity. In particular, this involves linking equipment, systems, and people in a networked production environment using technologies such as artificial intelligence (AI), the internet of things (IoT), and big data [7, 8, 9].

Indeed, AI has been successfully applied to a large number of industrial processes. For instance, the textile industry has used computer vision (CV) for both cloth inspection and defect detection throughout the production process [10, 11]. In the petrochemical industry, artificial neural networks (ANNs) have been used to recognize seismic patterns, and genetic algorithms have been utilized to optimize oil well performance [12]. The food industry has employed ANNs for food classification and production planning [13, 14, 15]. Last but not least, CV and other machine learning techniques have been used for flaw detection [16, 17] and preventive maintenance [18] in the metallurgical industry.

This master's thesis focuses on the application of AI to optimize energy consumption in one of the most sophisticated and well-known metallurgical processes: wire rod steel rolling. Wire rod steel rolling is an extremely complex industrial process characterized by the number and diversity of subsystems involved and the complex interactions between them. The workpieces in this process are billets (see Figure 1.1), which are semi-finished steel products with polished

corners. The main advantage of billets is that, being semi-finished products, they are easier to handle and transport than raw steel, which facilitates their commercialization and use in production processes.

Figure 1.1: Steel billets.

Figure 1.2: Billet exiting GSW's induction furnace.

In wire rod rolling, two main energy-intensive phases can be distinguished: the heating phase and the rolling phase. During the heating phase, the billets undergo various heating treatments in an induction furnace, preparing them for subsequent deformation. Following the heating phase, the rolling phase takes place, where the hot billets are passed through a wire rod rolling mill. This mill reduces their cross-section, increases their length, and shapes them into wire rods through compression. Once the wire rods have cooled down, they undergo additional processes such as straightening and cutting to achieve the desired final product.

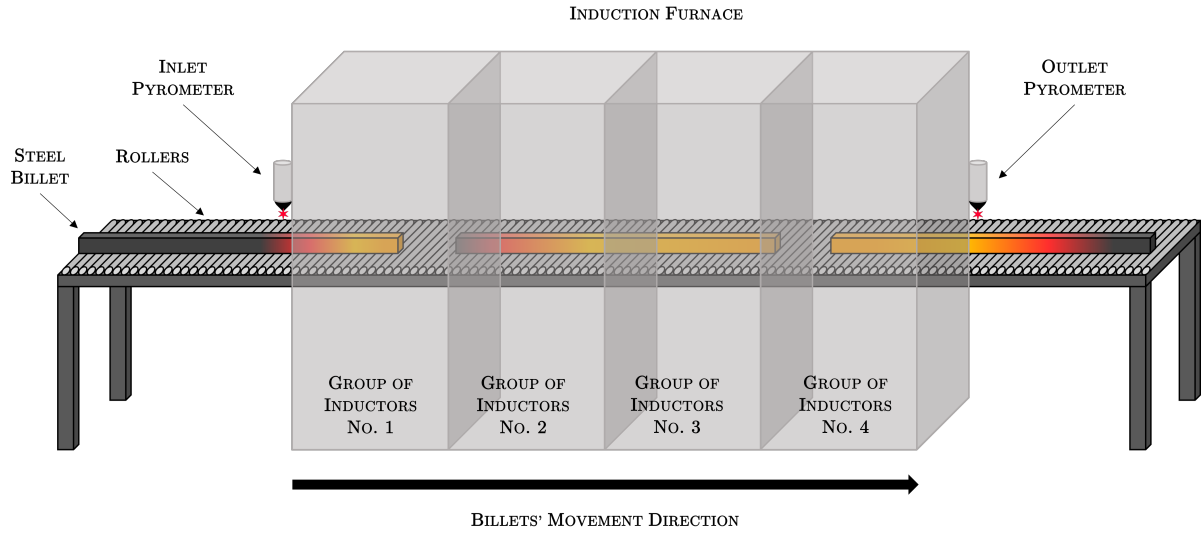


Figure 1.3: Sketch of the billet heating process in the induction furnace.

Specifically, the research conducted in this master's thesis concentrates on the heating phase. One of the key elements in this phase is the induction furnace (see Figure 1.2), which is used to heat the billets and reach the necessary temperature profiles for wire rod rolling, as mentioned

earlier. The furnace comprises a series of inductors that apply heat and rollers that are in charge of the billet's advance. Furthermore, the furnace is equipped with two pyrometers, one at the entrance and one at the exit, which record the billets' temperature (see Figure 1.3). Based on the input temperature and the target output temperature, the furnace regulates the power supplied by each of the inductors. However, in practice, the set target temperature is not always achieved, and the operator in charge has to manually adjust it. This provides leeway to seek more optimized and energy-efficient inductor management strategies from an AI-driven perspective.

The main objective of this thesis is to serve as a proof of concept for a methodology combining finite element (FE) modeling, machine learning (ML), and reinforcement learning (RL) aimed at optimizing the energy consumption of the induction furnace during the billet heating process. For this purpose, the work has been divided into three major phases:

- The first phase involves developing and validating a FE digital twin that simulates the heating process of a billet as it passes through the induction furnace.
- The second phase aims to develop a ML model based on ANNs that can generate real-time predictions of the temperature profiles of the billets at furnace's exit, leveraging the results from the previous FE model.
- The third phase focuses on establishing a RL framework able to optimize the energy efficiency of the induction furnace, specifically identifying inductor heating patterns that cut down on energy consumption. To this end, the ML model developed in the second phase serves as the virtual environment for the RL framework.

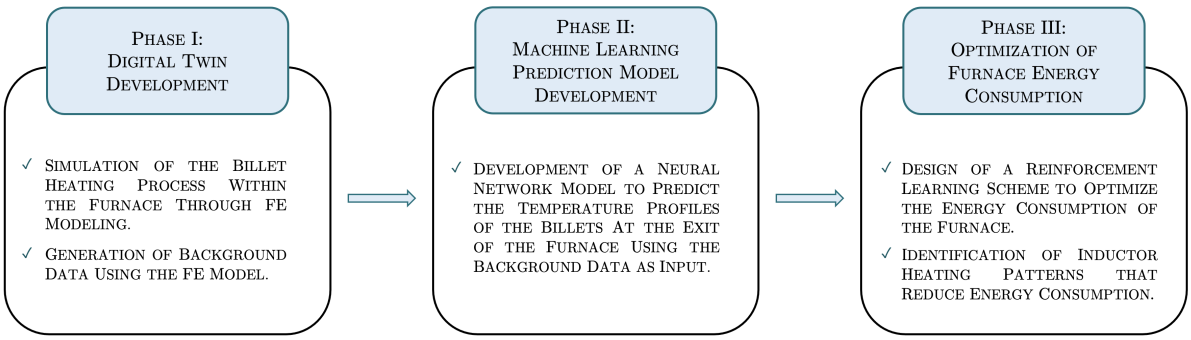


Figure 1.4: Outline of the methodology to be applied in the project.

To the best of our knowledge, no prior study has explored the application of AI to optimize energy consumption in the wire rod steel rolling process using our approach (summarized in Figure 1.4). Therefore, the methodology presented in this thesis not only opens up new possibilities for research but also holds potential for broader application, including other industrial processes such as glass manufacturing, cement production, or petroleum refining.

The remainder of this master's thesis is organized as follows: Chapter 2 provides an overview of the dataset and the preprocessing steps, as well as an explanation of the ML and RL techniques used throughout the project. Chapter 3 focuses exclusively on the FE digital twin of the

induction furnace. Chapter 4 delves into the development of a ML model based on ANNs to accurately predict the temperature profiles of the billets at the furnace's exit. Chapter 5 is dedicated to introducing a RL framework in order to optimize the energy consumption of the furnace during the billet heating phase. Finally, Chapter 6 concludes the thesis by presenting the conclusions and future work.

In order for this research to comply with FAIR (findable, accessible, interoperable, and reusable) principles, especially in terms of accessibility and reusability, both the dataset and the code developed for this thesis have been made openly available under an MIT license in the following GitHub repository:

https://github.com/bolivars/furnace-energy-optimization.

Chapter 2

Dataset and Methods

The purpose of this chapter is twofold: firstly, to provide a complete description of the dataset under study, including its origin and the applied preprocessing steps; and secondly, to offer some background on the ML and RL techniques used in this research.

It should be noted that all the tasks carried out in this thesis, including data importation, preprocessing, and the application of ML and RL techniques, have been performed using Python, version 3.9.16, and some of its libraries, including NumPy [19], Pandas [20], Matplotlib [21], and Keras [22], among others. Specific details regarding the libraries used in each project phase can be found in the headers of the Jupyter notebooks developed for this thesis, which are available in the GitHub repository.

2.1 Dataset Overview

This research has been conducted in partnership with Global Steel Wire (GSW), a steel manufacturer situated in Santander (Cantabria, Spain), which has provided the data required for the project. Specifically, the induction furnace under study in this thesis is located within the facilities of this steelmaker. It is worth mentioning that this furnace has dimensions of 20 meters in length, 2 meters in width, and 1 meter in height. Furthermore, together with the inlet and outlet pyrometers employed for monitoring the temperature of the billets during the heating phase, the furnace incorporates 12 induction coils arranged in series and grouped into 4 groups of inductors (each of them encompassing 3 coils). Most of the data provided by GSW corresponds to measurements related to one of the aforementioned devices.

The dataset comprises information concerning the heating process of 217 steel billets. For each of these billets, the following data is available:

- Identification and descriptive details such as the billet's unique identifier, length, and corresponding family (resulfurized, stamped, or microalloyed, among others).
- Multivariate time series data related to the heating process, including the billet's tip position in the induction furnace, the instantaneous power of each of the four groups of inductors, and the temperature measured by the inlet and outlet pyrometers. In total, there are seven time series available for each billet.

Figure 2.1 displays as an example the multivariate time series associated with the heating process of a specific billet. The subplots in the first two rows illustrate the instantaneous power of the four groups of inductors, while the left subplot in the third row represents the temperature monitored by the inlet and outlet pyrometers. The right subplot in the third row illustrates the position of the billet's tip as it passes through the furnace.

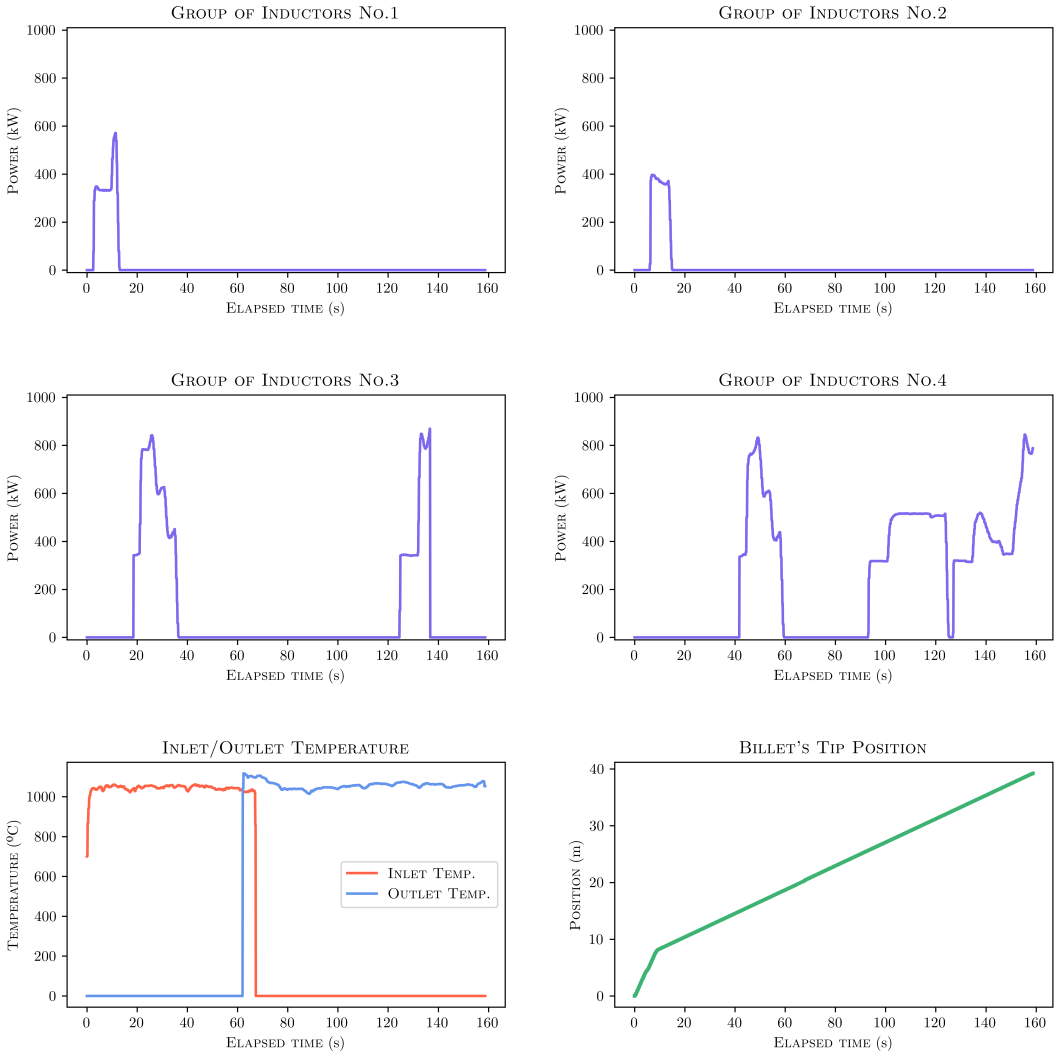


Figure 2.1: Multivariate time series associated with the heating process of billet 6211341.

The raw data provided by GSW, which consisted of CSV (comma-separated values) files, has undergone some preprocessing prior to modeling and analysis. In addition to typical preprocessing steps such as adapting the data formats for easier reading, other more specific actions that are worth highlighting have been carried out, namely:

- One issue was that, as billets are typically fed into the furnace sequentially (one after the other) to maximize energy efficiency, some time series contained recordings from different billets, causing overlap. For instance, this led to non-zero measurements at the outlet pyrometer even before the billet had passed through the furnace, which makes no sense.

To address this, inconsistent records were removed based on a criterion using the length of the billets.

- The granularity of the time series' temporal resolution was adjusted. Specifically, time series originally had a fine resolution of 0.1 seconds, but were resampled to a coarser period of 1 second to alleviate data processing.
- Outliers were also detected in the time series of the billet's tip position, potentially due to debris near the pyrometer. These outliers were corrected using interpolation techniques.
- The time each billet spent inside the furnace was variable depending on the billet's length and the heating treatment applied, which presented a consistency problem for models that required fixed-dimensional inputs, such as the FE model or ANNs. To solve this, the time series were extended by padding them with zero values until a predetermined fixed length of 200 one-second time steps was reached. This length was chosen because none of the billets for which data was available were in the furnace for more than 200 seconds.

It is worth mentioning that all the preprocessing of the raw data provided by GSW is well documented in the Jupyter notebooks developed for the thesis, which can be found in the GitHub repository created for the project.

2.2 Artificial Neural Networks in a Nutshell

This section provides an overview of the basic aspects of ANNs without delving into many details, as they have been extensively covered during the master's degree. In particular, special emphasis is given to fully connected neural networks (FCNNs), long short-term memory (LSTM) recurrent neural networks, and convolutional neural networks (CNNs), as these are the most commonly used ANN architectures in subsequent chapters.

2.2.1 Foundations of Artificial Neural Networks. Fully Connected Networks

ANNs are a computational model inspired by the learning mechanisms observed in the human brain [23]. Indeed, ANNs are made up of interconnected nodes known as neurons, which mimic biological neurons. These nodes establish connections based on a specified network architecture in charge of governing the transmission of information, with each connection having a weight that indicates its importance within the network [24]. Thus, neurons are computational units that receive an input signal (usually from other neurons) and compute the output signal by means of a nonlinear function known as the activation function, which is applied to the weighted sum of the inputs. This weighting is done on the basis of the weights associated with each connection.

FCNNs, also known as dense neural networks, are one of the most popular architectures for ANNs, alongside CNNs and LSTMs. In FCNNs, neurons are arranged in layers so that every neuron in a given layer is connected to every neuron in the subsequent layer (feed-forward network). This layered structure, shared with other architectures, consists of the input layer (responsible for data acquisition), the output layer (dedicated to producing predictions), and the hidden layers (those between the input and the output layers).

Regarding activation functions, the most popular ones include sigmoid, hyperbolic tangent, rectified linear unit (ReLU), and softmax, among others [25]. The activation function of the output layer is chosen according to the problem (usually softmax for classification tasks and linear for regression tasks), but it is critical to notice that the activation functions employed in

hidden layers must be nonlinear; otherwise, the resulting model would become a linear regression. For instance, in the regression problem tackled in Chapter 4, ReLU has been employed as the nonlinear activation function for the hidden layers (which helps mitigate the vanishing gradient problem), while a linear activation function has been used for the output layer.

The most important characteristic of ANNs is their ability to learn. Just as external stimuli are essential for human learning, external stimuli in ANNs are provided by the training set, which contains input observations where the desired output is known. In this sense, learning in ANNs is supervised and occurs by updating the weights associated with the connections between neurons. Therefore, the purpose of the training process is to find the weights for each connection in the network in such a way that the predicted output is as close as possible to the desired output. The way to assess the optimality of the obtained weights is through the loss function, which in the case of a regression problem like the one addressed in this thesis is typically the mean squared error (MSE). More formally, if the training set consists of N observations, the goal is to solve the following minimization problem:

$$\underset{\boldsymbol{\omega}\in\mathbb{R}^p}{\text{Minimize}} \quad \text{MSE}(\boldsymbol{\omega}) := \frac{1}{N}\sum_{i=1}^{N}\Big(\theta_i(\boldsymbol{\omega}) - \hat{\theta}_i\Big)^2, \tag{2.1}$$

where $\boldsymbol{\omega}$ represents the weights of all possible connections in the neural network, p is the number of total connections in the network, $\theta_i(\boldsymbol{\omega})$ is the model's prediction for the i-th observation of the training set, and $\hat{\theta}_i$ is its target value.

In order to tackle the previous problem, an iterative process is followed in which the network weights are updated so that the loss function is reduced. Various techniques exist for weight updates, including gradient descent, mini-batch gradient descent, and stochastic gradient descent, along with their variants that incorporate momentum [26]. Specifically, the analysis in this thesis will be performed using *Adam* [27], an extension of gradient descent with momentum.

One of the challenges associated with ANNs is that they are prone to overfitting, leading to high variance. Overfitting occurs when a model fits the training set remarkably well, but fails to generalize to unseen data. Several techniques can be employed to address this issue, such as L1 or L2 regularization (which incorporates a term in the loss function accounting for the weight vector's norm), dropout (randomly deactivating nodes to simplify the network and prevent over-specialization), or data augmentation (to increase the diversity of the training data).

2.2.2 Long Short-Term Memory Recurrent Neural Networks

LSTM networks are a subset of recurrent neural networks (RNNs), which are ANNs with feedback connections enabling the processing of sequential inputs [28, 29]. LSTMs were mainly developed to address two common challenges that conventional RNNs face: (1) the vanishing gradient problem in deep architectures, and (2) their limited ability to learn and retain long-term dependencies.

The architecture of LSTMs is more complex than that of conventional RNNs, incorporating memory and gate mechanisms that supervise the flow of information through the network [30]. This complex design enables LSTMs to capture long-term dependencies in sequential data, making them particularly well-suited for diverse tasks, including but not limited to time series analysis (the focal point of this thesis) [31], natural language processing [32], and even speech generation [33]. The architecture of LSTM cells is explained in a little more detail below, with emphasis on the long-term recall mechanism.

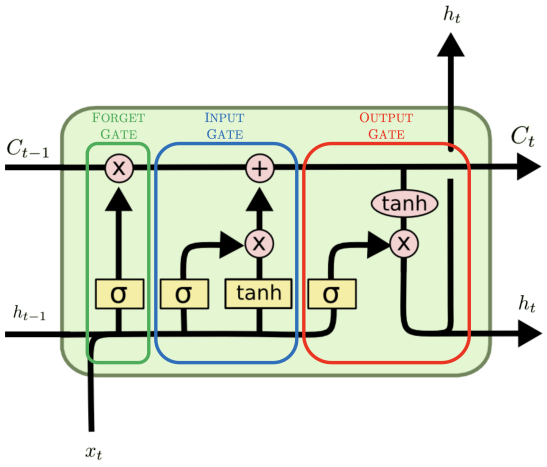
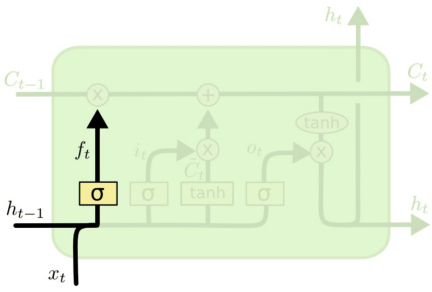
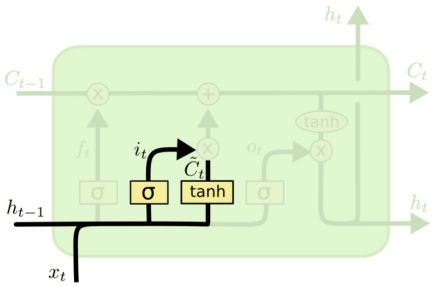


Figure 2.2: LSTM cell architecture (graph inspired in [34]).

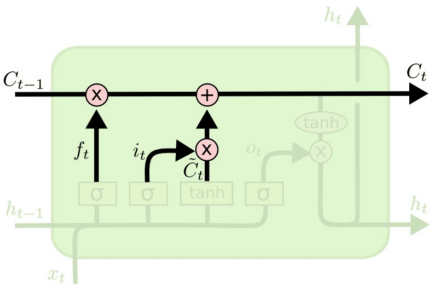
Figure 2.2 illustrates the architecture of an LSTM cell. The input at time t, denoted as x_t, is represented by a line towards the LSTM cell, while the output, h_t, is represented by an arrow leaving the cell. The LSTM unit is made up of four dense layers (three of them with sigmoid activation function, and one of them with hyperbolic tangent), which are shown as yellow rectangles. These layers help process the information as well as make decisions within the LSTM cell. In addition, the small pink circles represent point-wise operations between vectors in the LSTM cell. These operations allow the integration of information across the different components of the LSTM unit.



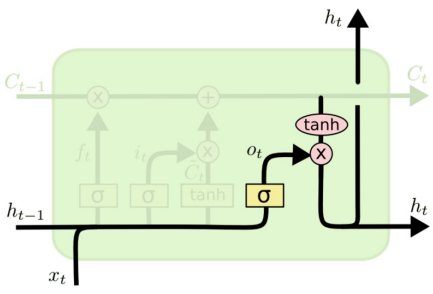
(a) Step I: Forget Gate.



(b) Step II: Input Gate.



(c) Step III: Cell State Update.



(d) Step IV: Output Gate.

Figure 2.3: Learning process in a LSTM cell [34].

The cell state, C_t, represented by the top horizontal line that spans the unit from left to right in Figure 2.2 and Figure 2.3, is probably the most important component of the LSTM unit. The cell state is in charge of controlling the information flow inside the network and, thus, of managing the recall of dependencies and/or features in the long term. Even though it often remains unchanged, it is feasible to delete (forget) or add (remember) information to the cell state in a controlled way using structures known as gates [34].

Figure 2.3 shows a diagram (taken from [34]) of the learning process in an LSTM cell. The first step (see Figure 2.3a) is to decide which information should be remembered and which should be forgotten, a task that is performed by the forget gate, f_t. This gate uses the sigmoid to assign a value in $[0, 1]$ to each element/feature in C_{t-1} based on the values of x_t and h_{t-1}. In particular, a value of 0 blocks the information flow for that particular feature (completely forget), whereas a value of 1 fully allows the information to flow (completely remember). Intermediate values weigh the significance of the information to be remembered. Formally, this is represented as

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right), \tag{2.2}$$

where W_f and b_f are the weights and bias, respectively, of the forget gate.

The second step (see Figure 2.3b) is to decide which new information should be stored in the cell state. This is accomplished through the input gate, denoted by i_t, which is a sigmoid layer that selects which components of the cell state should be updated. Furthermore, a hyperbolic tangent function generates a new state vector, $\hat{C}_t$, which can be added to the cell state. Mathematically, the above operations can be written as

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \text{ and } \hat{C}_t = \tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right), \tag{2.3}$$

where W_i, b_i, W_C and b_C represent the weights and biases associated with the input gate and the layer with hyperbolic tangent activation, respectively.

The third step (see Figure 2.3c) is to update the cell state by combining the results of the forget gate, the input gate, and the new information. Formally, this update can be expressed as

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t, \tag{2.4}$$

where the operation $*$ is an element-wise product.

The fourth step (see Figure 2.3d) is to determine the output, h_t, of the LSTM cell. This output is a filtered version of the cell state that is obtained through a sigmoid layer as follows:

$$o_t = \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right) \text{ and } h_t = o_t * \tanh\left(C_t\right), \tag{2.5}$$

where W_0 and b_0 are the weights and bias associated with the last sigmoid activation layer.

2.2.3 Convolutional Neural Networks

CNNs [35, 36] represent a specific type of ANNs in which consecutive layers are connected through a convolutional operation that enables them to learn complex spatial structures or patterns within data, especially data having a grid-like topology, such as images, videos, or even time series. Unlike FCNNs, CNNs are able to learn local patterns rather than global patterns from the input feature space. This is achieved by setting up the parameters in filters

or kernels that convolute across the dimensions of the input layer to extract features, allowing for local learning [37].

A typical CNN architecture is made up of stacked repetitions of convolutional blocks, which are in charge of feature extraction and are usually composed of a convolutional layer followed by a pooling layer for dimension reduction. Following the convolutional blocks, there is a flattening layer that converts the feature maps into a one-dimensional vector that serves as the input layer for one or more fully connected layers that handle the classification or regression task, depending on the problem at hand.

Because of their relationship with computer vision, 2D CNNs are often the first architecture that comes to mind when thinking about CNNs. In 2D CNNs, filters are small, two-dimensional windows typically convoluted with images or video frames. However, when dealing with one-dimensional time series, such as audio recordings, stock prices, or electrocardiogram signals, 2D CNNs become incompatible from a dimensional point of view. As a consequence, 1D CNNs were introduced, which have been shown to be particularly useful for the analysis of 1D signals [38]. The main difference with a 2D CNN is that the filters are one-dimensional, so there are fewer parameters, which reduces computational complexity and eases training and implementation.

The data available for this research are multivariate time series, as explained in Section 2.1, since several variables (instantaneous power of the four groups of inductors, advance of the tip of the billet inside the furnace, temperatures...) are recorded as a function of time. Contrary to what might be thought, this data structure can fit perfectly with a 1D CNN scheme: it is enough that the input is a one-dimensional time series with several channels, each of them accounting for one of the variables of interest (similar to when a color image is divided into three channels: red, green, and blue (RGB)). Figure 2.4 shows the 1D CNN architecture used as the basis for the convolutional models implemented in this thesis.

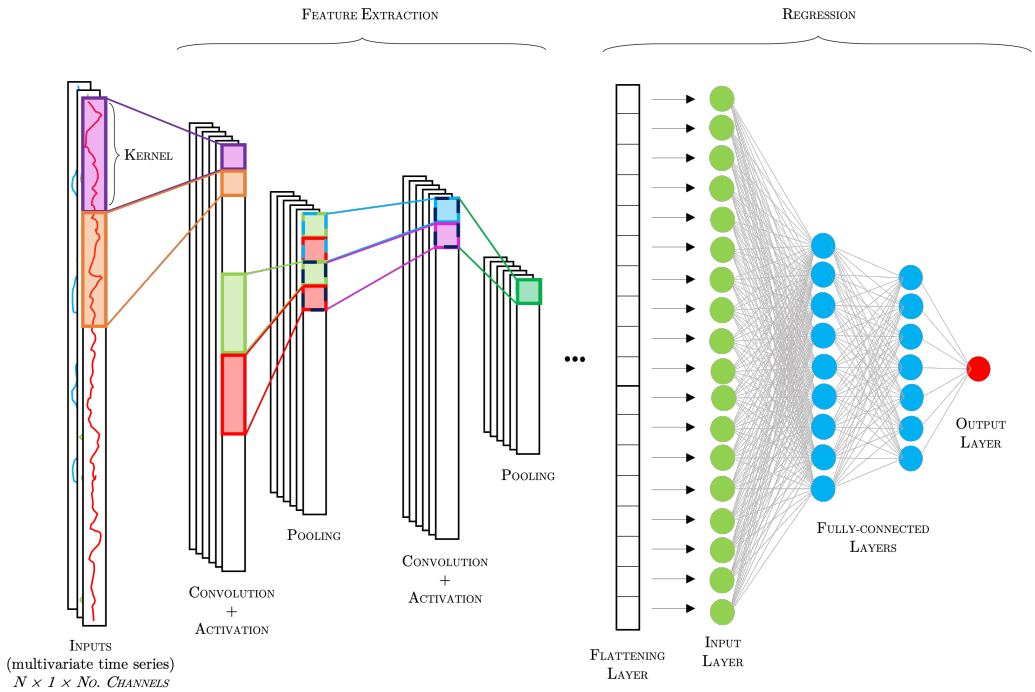


Figure 2.4: One-dimensional convolutional neural network architecture for a regression problem.

2.3 Background on Reinforcement Learning

RL is one of the three learning paradigms of ML, together with supervised and unsupervised learning. RL is a framework where an agent learns to make decisions based on its interactions with a dynamic environment, with the goal to maximize a reward [39]. Specifically, the decision-making policy is learnt through trial and error [40]. Hence, there are two key elements in RL: the environment (representing the problem to be solved) and the agent (representing the learning algorithm).

The agent constantly interacts with the environment, so that at each time step t of an episode, the agent makes a decision by taking an action a_t based on its policy, $\pi(s_t)$, being s_t the state of the environment at time t. In response to this action, the state of the environment transitions to an updated state, s_{t+1}, and the agent receives a reward, r_{t+1} [39]. The agent's ultimate goal is to update its decision-making policy to maximize the discounted cumulative sum of rewards in an episode, R, which is defined as follows:

$$R := r_0 + \gamma^1 r_1 + \ldots + \gamma^t r_t + \ldots + \gamma^T r_T := \sum_{t=0}^{T} \gamma^t r_t, \tag{2.6}$$

where r_t is the reward at time step t, T is the time-step at which the episode terminates, and $\gamma \in [0, 1)$ is the discount factor. Given that $\gamma < 1$, this factor is in charge of discounting future rewards relative to immediate rewards [39].

There are plenty of approaches to tackle RL problems, including policy gradient methods [41], actor-critic methods [42], Monte Carlo methods [43], temporal difference methods [44], and Deep Q-Networks [45], among others. Nevertheless, this thesis only focuses on Deep Q-Networks, which are explained in greater detail below.

2.3.1 Deep Q-Networks

In 2015, DeepMind developed Deep Q-Network (DQN) [45], an algorithm capable of solving a wide variety of Atari games by combining a classical RL technique, Q-learning, and deep neural networks (DNNs). The development of DQNs set a precedent for laying the groundwork for what is nowadays known as Deep Q-Learning. Before diving deeper into it, a brief overview of Q-Learning is given to become familiar with the terminology.

Q-Learning

Q-learning (QL) [39, 46] is a classical RL technique based on a state-action function known as the Q-function. Given a decision-making policy π, the Q-function $Q^\pi(s_t, a_t)$ represents the expected discounted cumulative sum of rewards that would be obtained from a state s_t by first taking an action a_t and then following the policy π. Formally,

$$Q^\pi(s_t, a_t) = \mathbb{E}[R_t | s_t, a_t, \pi], \tag{2.7}$$

where $R_t := \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$ is the discounted cumulative sum of future rewards from time t [45].

Consequently, the optimal Q-function $Q^*(s_t, a_t)$ is defined as the maximum expected cumulative reward that could be obtained by starting from the state s_t, taking the action a_t, and following any policy π. The expression for the optimal Q-function obeys Bellman's equation [39], that is,

$$Q^*(s_t, a_t) = \max_\pi \mathbb{E}[R_t | s_t, a_t, \pi] = \mathbb{E}[r_{t+1} + \gamma \max_{a'} Q^*(s_{t+1}, a')]. \tag{2.8}$$

In other words, the previous identity indicates that the optimal Q-value of an action a_t applied to a state s_t, $Q^*(s_t, a_t)$, is the immediate reward (the one obtained after applying the action a_t to state s_t) plus the maximum expected reward that could be obtained in the next state s_{t+1} corrected by the discount factor γ.

The idea behind QL is to approximate $Q^*(s_t, a_t)$ by means of an iterative process, as it can be ensured that $Q_i \longmapsto Q^*$ as $i \longmapsto \infty$ [39]. The iterative update [47] can be expressed as follows:

$$Q_{i+1}(s_t, a_t) \leftarrow (1-\alpha)Q_i(s_t, a_t) + \alpha\left(r_{t+1} + \gamma \max_{a'} Q_i(s_{t+1}, a')\right), \tag{2.9}$$

where $\alpha \in (0, 1]$ is the learning rate, and r_{t+1} is again the reward obtained when transitioning from state s_t to state s_{t+1}. Therefore, the Q-function is nothing more than a table that stores the "optimal" Q-value for each action-state pair obtained during the iteration process.

Deep Q-Learning

One of the main disadvantages of QL is that each action-state pair must be visited many times to obtain a reliable approximation of the Q-values via the iterative process. Consequently, its application is limited to problems with discrete, finite, and few-element action and state spaces [46]. However, in most real-world problems, the state space is very large, even continuous. One possible solution to overcome this problem is to use a function that approximates the Q-function, instead of using a lookup table as QL does. Indeed, finding a parametrized function $Q(s, a; \theta) \approx Q^*(s, a)$ that approximates the optimal Q-function, not only allows to obtain the Q-values quickly (lookup tables are sometimes not efficient), but also enables to estimate Q-values for unseen states [39, 46, 45].

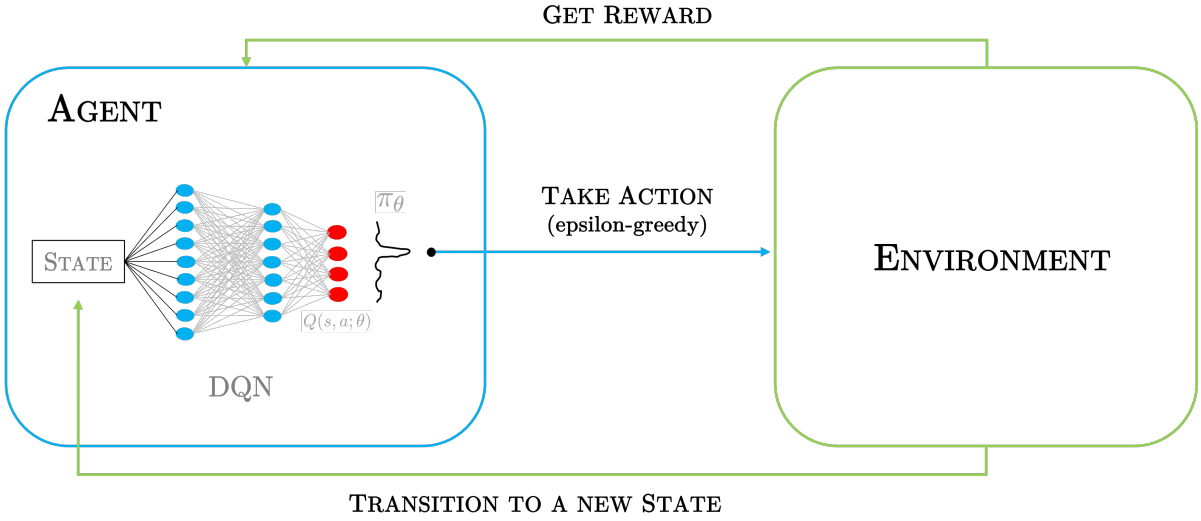


Figure 2.5: Structural diagram of Deep Q-Learning.

In this context, DQL [48, 49] emerges as a RL method that approximates the Q-function using a ANNs. Specifically, DQL uses DQNs to approximate the Q-function, allowing it to capture more complex patterns. Figure 2.5 shows a classic DQL diagram, where the most remarkable aspect is that the agent is now a DQN. Furthermore, it is observed that the states are the

inputs to the DQN, while the outputs correspond to the predicted Q-values (each neuron in the output layer corresponds to an action in the action space). The DQN's parameters are iteratively updated in order to minimize the following loss function [45]:

$$L_i(\theta_i) := \mathbb{E}\left[\left(r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a', \theta_{i-1}) - Q(s_t, a_t, \theta_i)\right)^2\right], \tag{2.10}$$

where θ_i denotes the parameters of the DQN at the i-th iteration.

In light of the above, in order to apply DQL it is necessary to: define the space of actions and states, fix the learning objective (in the case of chess, for instance, would be capturing the king), establish the reward system, and determine the initial state from which actions can be taken.

How are actions selected?

Once the DQN has been trained, what is obtained is a decision-making policy π_θ, where θ are the parameters of the DQN. Thus, given a state s, the optimal action, a^*, is the one that maximizes the obtained Q-value following the learned policy, that is,

$$a^* = \text{argmax}_a \, Q(s, a; \theta). \tag{2.11}$$

Consequently, by taking the optimal action, a^*, it is ensured that the environment transitions to a new state that maximizes the reward.

However, it is common to use an epsilon-greedy action selection criterion to ensure good coverage of the state-action space (something that is sometimes not achieved if the optimal policy is strictly followed). This epsilon-greedy algorithm balances the exploration and exploitation of the environment by randomly choosing between exploration and exploitation [40]. In particular, this method exploits the learnt decision-making policy most of the time, with a small probability of randomly exploring [50]. According to this criterion, the selected action at time t would be

$$a_t = \begin{cases} \text{argmax}_a \, Q(s_t, a; \theta), & \text{with probability } 1 - \varepsilon. \\ \text{choose an action at random}, & \text{with probability } \varepsilon, \end{cases} \tag{2.12}$$

where ε is the exploration probability.

Chapter 3

Development of a Digital Twin of the Induction Furnace

This chapter focuses on the initial phase of the project, whose objective is to develop and experimentally validate a FE-based digital twin of GSW's induction furnace that faithfully simulates the heating process of the billets. The chapter provides a description of the FE model, including its purposes, fundamental assumptions, and implementation details. Furthermore, it outlines the calibration process of the FE model and assesses the accuracy of the simulations.

3.1 Implementation of the Finite Element-Based Digital Twin

Examining the influence of operating conditions on the performance of GSW's induction furnace is of great interest for two reasons. Firstly, it facilitates the analysis of the final product's quality from a metallurgical point of view. Secondly, it offers a chance to explore innovative inductor management strategies aimed at reducing energy consumption during the heating phase. However, a major challenge in pursuing these objectives is the lack of data. This is because the primary focus of production laboratories, such as GSW's, is not data generation for analysis, but the manufacturing of products for commercialization. Therefore, the need arises to develop and use a digital twin of the induction furnace, i.e., a virtual model of the furnace that, receiving the same inputs as the real furnace, is able to faithfully reproduce its output.

In this context, mathematical modeling using finite elements proves to be a valuable asset for developing the digital twin. By incorporating all relevant physical laws, including those related to heat transfer during the heating phase, FE modeling allows to create a virtual prototype of the furnace that closely mimics GSW's real furnace behavior. This characteristic makes the FE-based simulator a robust model with excellent extrapolation capabilities, thereby enabling simulations of various operating conditions (especially different heating patterns), which may not be feasible in GSW's factory. Consequently, the FE-based digital twin becomes a powerful tool for generating background data that can be directly analyzed or used to train a ML model capable of making real-time predictions of the temperature profiles of the billets upon exiting the furnace, as discussed in Chapter 4. It is beyond the scope of this thesis to tackle the theoretical foundations of FE numerical modeling; please refer to [51, 52, 53] for a detailed mathematical description.

3.1.1 Finite Element Model Overview

A FE model has been developed to simulate, in a simplified manner, the heating process of steel billets as they pass through GSW's induction furnace. Indeed, the model successfully predicts the temperature profiles of the billets as they exit the furnace. It is worth highlighting that the term "temperature profile" refers to the time series that would be measured by the outlet pyrometer along a horizontal line on the upper surface of the billet as it leaves the furnace (see Figure 3.3).

The FE model uses five time series as inputs to simulate the temperature profile of each billet upon exiting the furnace. These include the four time series corresponding to the instantaneous power of each of the four groups of inductors (also referred to as heating pattern), and the time series representing the billet's position or advance within the furnace. It is worth noting that the time series used to feed the model have been preprocessed, rather than using the original raw data, according to the criteria outlined in Section 2.1.

Furthermore, the model imposes two boundary conditions in addition to those dictated by the furnace's real geometry. The first condition is the initial temperature of the billets, which is determined by averaging the time series measured by the inlet pyrometer. The second condition is the temperature of the air surrounding the furnace, which is supplied by GSW and allows to take into account the heat convection processes between the air and the steel. Lastly, the model incorporates some thermal properties of steel, such as conductivity, specific heat, density, and air-steel convection coefficient, which were obtained from the literature.

3.1.2 Finite Element Model Assumptions

It is important to bear in mind that the developed FE model simplifies the heating process of the billets. Consequently, some physical processes are simplified, including:

- The energy exchange by radiation is replaced by an equivalent convective process, which considerably simplifies the simulations because radiation is a highly non-linear process.
- The heat density generated internally in the steel due to induction is assumed to be uniform in each cross section, which is not strictly true since theoretical induction models indicate that the distribution of the thermal intensity generated depends on the electromagnetic frequency of the induction.
- Inductor losses, which are non-linear and depend not only on the material used but also on the working power of the inductors, have not been taken into account. However, only a fraction of the energy consumed by the inductors is actually invested in heating the materials, so it would be ideal to upgrade the existing FE model to include an electromagnetic FE model that accounts for these losses.
- The temperature of each billet at the entrance of the furnace is assumed to be constant across its entire surface, with the assigned value being the average of the time series measured by the inlet pyrometer. However, small variations in the initial surface temperature distribution around this mean value are not expected to have a significant effect on the temperature profile at the furnace exit, which makes it a reasonable assumption.

3.1.3 Implementing the Finite Element Model using ANSYS MAPDL

The FE model was implemented using Mechanical ANSYS Parametric Design Language (MAPDL), a programming language designed for 3D modeling using FE methods [54]. Initially, simulations were conducted using the native applications developed by ANSYS (the software provider). However, they ended up being performed using the PyMAPDL module [55] of the PyAnsys library, which enables the usage of ANSYS products through Python [56]. The main reasons for selecting this Pythonic interface were its interoperability (allowing for the integration of data preprocessing and simulations within a single script) and its support for parallel computing. The latter made it possible to reduce the simulation time for each billet to 7 minutes, compared to the 20 minutes required when using the native applications.

Figure 3.1 shows an example of a three-dimensional simulation of a billet employing PyMAPDL, which, as mentioned above, supports Pythonic access to MAPDL. The visualization includes a color map showing the temperature distribution over the surface of the meshed billet.

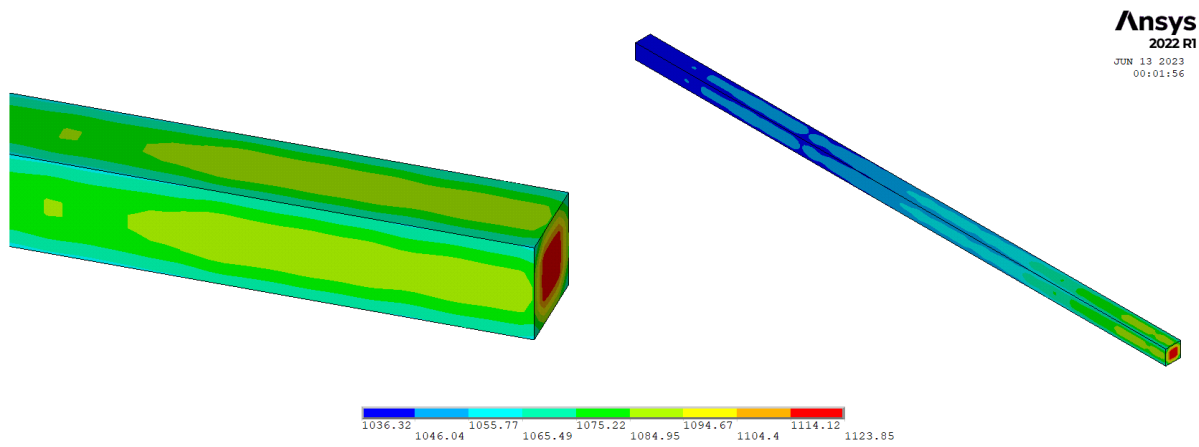


Figure 3.1: Three-dimensional simulation of a billet.

To obtain a more detailed description of the implementation of the FE model, including the definition of meshed surfaces and volumes, and the application of loads, please refer to the Jupyter notebook responsible for the FE simulation, which can be found in the GitHub repository.

3.2 Calibration and Analysis of Digital Twin Outcomes

The FE model has undergone a calibration process to fine-tune the simulation parameters (especially the one controlling the equivalent convection), ensuring that the simulations replicate the behavior of GSW's induction furnace as accurately as possible. In this calibration process, the reference values were the time series corresponding to the billet temperature profiles at the furnace exit recorded by GSW. Specifically, the main objective of the calibration was to achieve the maximum possible resemblance between the temperature profiles simulated using the FE model and the temperature profiles measured by GSW's outlet pyrometer. Figure 3.2 represents a flow chart of the calibration procedure.

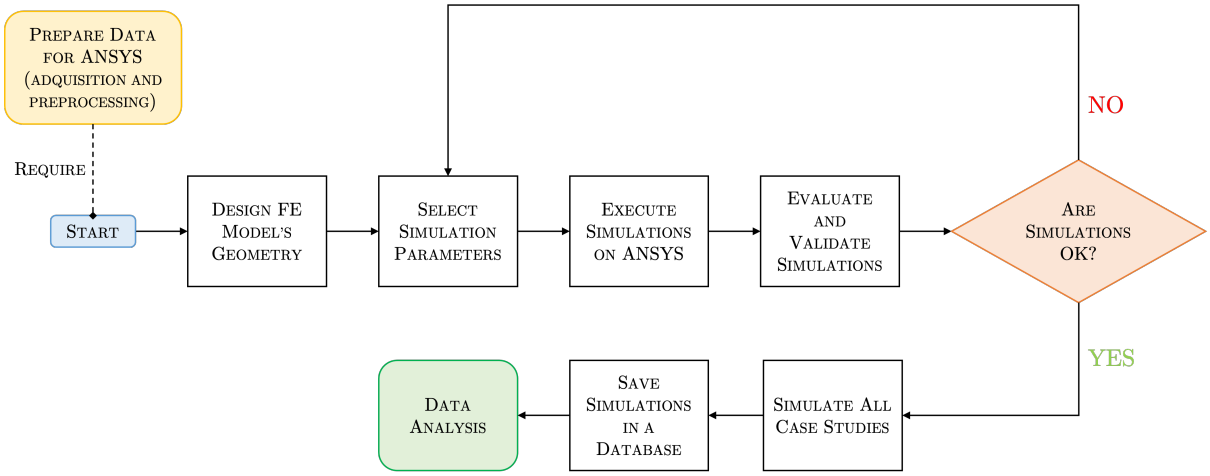


Figure 3.2: Flow chart of the calibration procedure.

The metrics used to assess the quality of the simulations were the Root Mean Squared Error (RMSE) and the Pearson correlation coefficient, ρ, which are defined as follows:

$$\text{RMSE}(T, T^s) := \sqrt{\frac{1}{n}\sum_{i=1}^{n}(T_i^s - T_i)^2}, \text{ and} \tag{3.1}$$

$$\rho(T, T^s) := \frac{\sum_{i=1}^{n}\left(T_i - \overline{T}\right)\left(T_i^s - \overline{T^s}\right)}{\sqrt{\sum_{i=1}^{n}\left(T_i - \overline{T}\right)^2 \sum_{i=1}^{n}\left(T_i^s - \overline{T^s}\right)^2}}, \tag{3.2}$$

where $T \in \mathbb{R}^n$ is the temperature profile of the billet upon exiting the furnace measured by GSW's outlet pyrometer, and $T^s \in \mathbb{R}^n$ is the temperature profile simulated using the FE model.

It is worth noting that RMSE is a measure of accuracy, while the Pearson correlation coefficient is a measure of association. Indeed, RMSE evaluates the point-to-point correspondence between simulated and observed time series values, with lower values indicating a better fit. On the other hand, the Pearson correlation coefficient measures the linear relationship between measured and simulated time series values, having a score between -1 (perfectly negative correlation) and 1 (perfectly positive correlation).

For instance, Figure 3.3 displays the temperature profile for the steel billet with ID 6211341, whose time series were previously illustrated in Figure 2.1. The temperature profiles shown correspond to the measured and/or simulated temperatures along a central horizontal line on the billet's surface (measurements are avoided at the edges due to loss effects).

Initially, it was intended to calibrate the FE model so that it could reproduce the valleys or cold spots observed in the black curve of Figure 3.3. These valleys occur due to the contact between the billets and the rollers that make them advance, resulting in heat losses. However, it was finally decided to calibrate the model to replicate the trend. The reason is that GSW was not interested in the exact positions of the cold spots but rather in the presence of consistent temperature drops over time, something that is reflected in the trend. It is worth mentioning

that the FE model was exclusively calibrated using billets belonging to the resulfurized family, as data related to other billet families was provided in subsequent phases of the project.

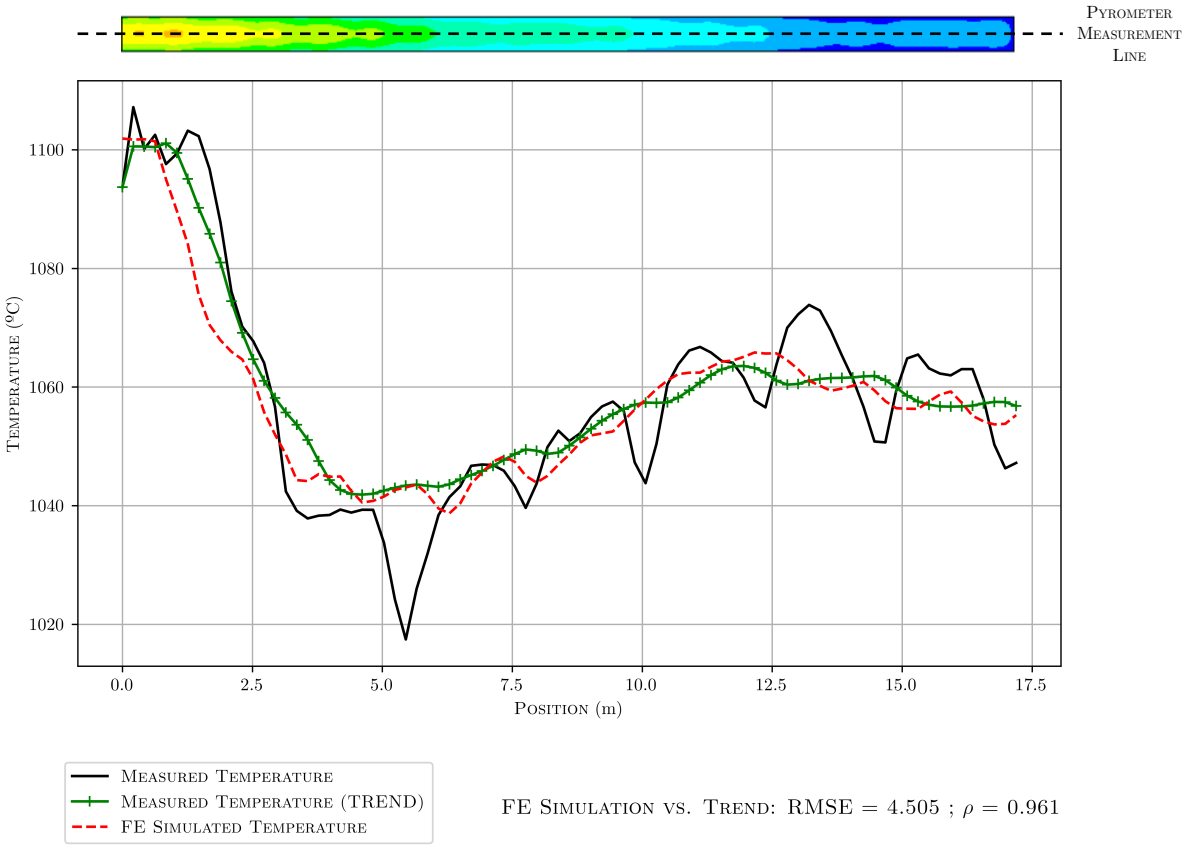


Figure 3.3: Comparison between the temperature profile measured at the exit of GSW's induction furnace and the one simulated using the simplified FE model for billet 6211341. The black curve represents the temperature profile measured by the outlet pyrometer, while the green curve illustrates its trend (obtained through exponential moving averages). The red curve represents the temperature profile simulated using the FE model. Evaluation metrics are included in the lower right corner.

Continuing with the example presented in Figure 3.3, the numerical results demonstrate a situation in which an RMSE of 4.505 and a Pearson correlation coefficient of 0.961 are obtained between the simulated temperature profile (red curve) and the trend of the temperature profile measured by GSW's outlet pyrometer (green curve). It is noteworthy that the RMSE represents the average point-to-point difference between the simulated and actual values, expressed in degrees Celsius in this case. Considering that the measurement error of the output pyrometer fluctuates between 10ºC and 20ºC depending on the temperature, it can be concluded that the simulated profile aligns quite accurately with the measured profile.

The aforementioned graph and discussion belong to a specific billet, the one with ID 6211341, which might suggest that the favorable results obtained are an isolated case. However, this behavior has been observed consistently across the entire family of resulfurized billets, as illustrated in Figures 3.4 and 3.5. These two histograms show the distribution of the RMSE and the

Pearson correlation coefficient between the simulated temperature profiles and the measured trends within the resulfurized billet family. It is observed that all RMSE values are below 5^{o}C, and all correlations exceed 0.95, thus highlighting the accuracy of the FE model.

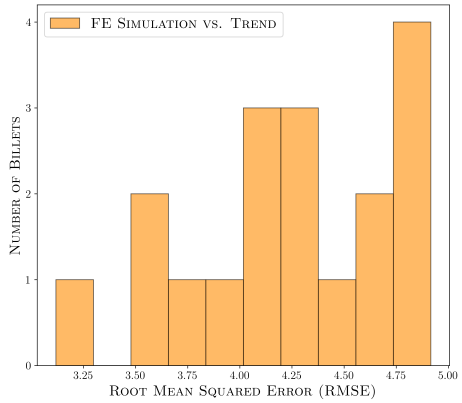


Figure 3.4: Histogram displaying the distribution of the RMSE for billets belonging to the resulfurized family.

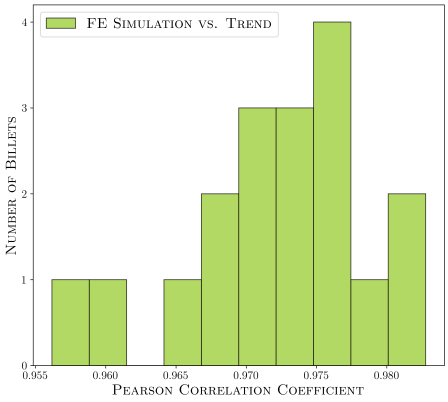


Figure 3.5: Histogram displaying the distribution of the correlation for billets of the resulfurized family.

In light of the obtained results, it can be concluded that the simplified FE model is a valuable tool for accurately predicting the temperature distributions of the billets upon exiting GSW's induction furnace based on the operating conditions (the time series corresponding to the advance of the billet and the instantaneous power of the four groups of inductors). This information regarding the billets' temperature distribution is very useful from a metallurgical point of view to determine the quality of the material and optimize subsequent processes involved in wire rod rolling.

Chapter 4

Using Neural Networks to Exploit the Digital Twin

This chapter delves into the second phase of the project, which seeks to develop some ML models based on ANNs that leverage the outcomes of the FE model to generate real-time predictions of the temperature profiles of the billets at the induction furnace's exit. The chapter discusses the need and importance of these ML models, outlines their architecture, describes the methodology employed for implementing them, and presents the obtained results.

It is worth mentioning that these ANN-based models have been implemented using Keras library (version 2.10.0) in Python [22]. Specifically, the training process has been boosted using a NVIDIA GeForce RTX 3090 GPU.

4.1 The Role of the Machine Learning Models

Thus far, an accurate FE model has been developed, showing remarkable extrapolation capabilities and robustness as it encapsulates most of the physical laws governing the billet heating process. However, one limitation of this FE model is the simulation time, especially when considering its implementation in a factory as part of an intelligent decision-making algorithm. Indeed, the simulation time of the FE model is approximately 7 minutes, nearly three times the time each billet spends inside the furnace. This makes the FE model unsuitable for real-time predictions, thereby reducing its practical utility.

In this context, training a ML model to predict the temperature profiles emerges as a highly feasible solution since, once trained, its evaluation in unseen data is extremely fast, taking only a few milliseconds. Furthermore, the quick evaluation of ML models makes them appropriate to be used within a RL framework, as will be seen in Chapter 5.

Indeed, RL consists of a trial-and-error process involving thousands of iterations, as mentioned earlier in Section 2.3. In each iteration, the current state transitions to a new state based on the selected action, which makes it necessary to have a method capable of quickly computing the reward to assess the decision that has just been made. In the problem under study, the reward will depend mainly on the RMSE between the target and the simulated temperature profiles, as will be explained later in Chapter 5. Therefore, in each iteration, it is necessary to simulate the billet temperature profile associated with the heating pattern reached after applying the

selected action. Consequently, this methodology is only feasible if a fast simulator such as an ML model is available, but would be impractical for the FE model because of its long simulation time.

In light of the above, it may seem that a time-consuming FE simulator is useless. However, even though it might not be useful to be integrated into an RL scheme, it is a valuable tool for generating synthetic samples thanks to its robustness and extrapolation capability (a task for which most ML models are not competitive because, although they interpolate very well, they are not able to correctly extrapolate). Indeed, it is enough to use Monte Carlo techniques to generate synthetic realistic operating conditions (that is, time series representing the advance of the billet inside the furnace and the instantaneous powers of the inductors), and feed them into the FE simulator to obtain the associated temperature profiles. This is particularly useful in the industrial sector, where one of the limitations when applying AI is the lack of data: most factories focus on production, which limits the number of real samples available for research. However, since this thesis is a proof of concept aiming to test the proposed methodology and synthetic data is not yet available, the idea of increasing the dataset by means of these synthetic samples is left for future work.

4.2 Implementation and Results of the Neural Network Models

Two ML models were developed using recurrent and/or convolutional neural networks. These models were trained using data from 200 out of the 217 available steel billets, allocating the remaining 17 (approximately 8%) to the test set. It is important to note that these are regression models designed to predict the temperature profile of the billets at the exit of the furnace. Hence, the target variable is represented by a temperature vector that captures the temperature profile.

On the one hand, the inputs of the models were the same as those used by the FE model, namely: the preprocessed time series of the advance of the billets inside the furnace and the instantaneous power of the four groups of inductors (a multivariate time series with 5 channels). On the other hand, the target vectors of the training set observations corresponded to the temperature profiles simulated via the FE model (not the time series registered by the outlet pyrometer).

Even though the time series related to the temperature recordings of the outlet pyrometer were available for the training set observations, the simulated temperature profiles were used as the target vectors to train the ML models for two main reasons. Firstly, to mitigate the influence of cold spots, which are present in the outlet pyrometers' recordings but are not replicated by the FE model (it is recalled that its purpose is to replicate the overall trend). Secondly, when in later phases of the project synthetic samples are used to retrain the ML models, it will not be possible to compare the predicted temperature profiles with real time series as they will not be available, which makes it necessary to exclusively work with the simulated time series.

It is worth mentioning that in order to reduce the complexity of the problem and avoid a large number of neurons in the output layer, the temperature profiles have been interpolated to only 50 points, despite the original simulated profile having 200 points. Figure 4.1 shows an example of the original simulated profile with 200 points compared to the interpolated profile with 50 points. Although some fine details are lost, the overall trend is preserved.

In the following, the implemented models are described, and the results obtained with each of them are discussed. Once again, the RMSE was used as the evaluation metric to assess the performance of the models.

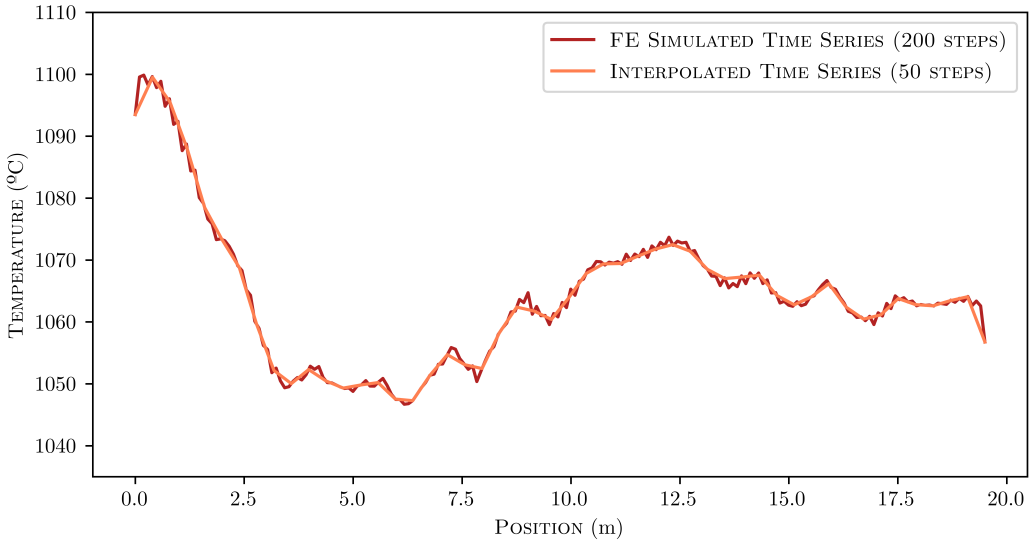


Figure 4.1: Interpolation of the simulated temperature profiles to reduce the models' complexity.

4.2.1 One-Dimensional Convolutional Model

This neural network model, as its name suggests, is mainly based on one-dimensional convolutional layers (see Section 2.2.3). The architecture of the model, depicted in Figure 4.2, consists of three convolutional blocks, each one composed of a 1D convolutional layer with ReLU activation, a batch normalization layer, and a pooling layer. These blocks are followed by a flattening layer with dropout, and two dense layers with L2 regularization to mitigate overfitting.

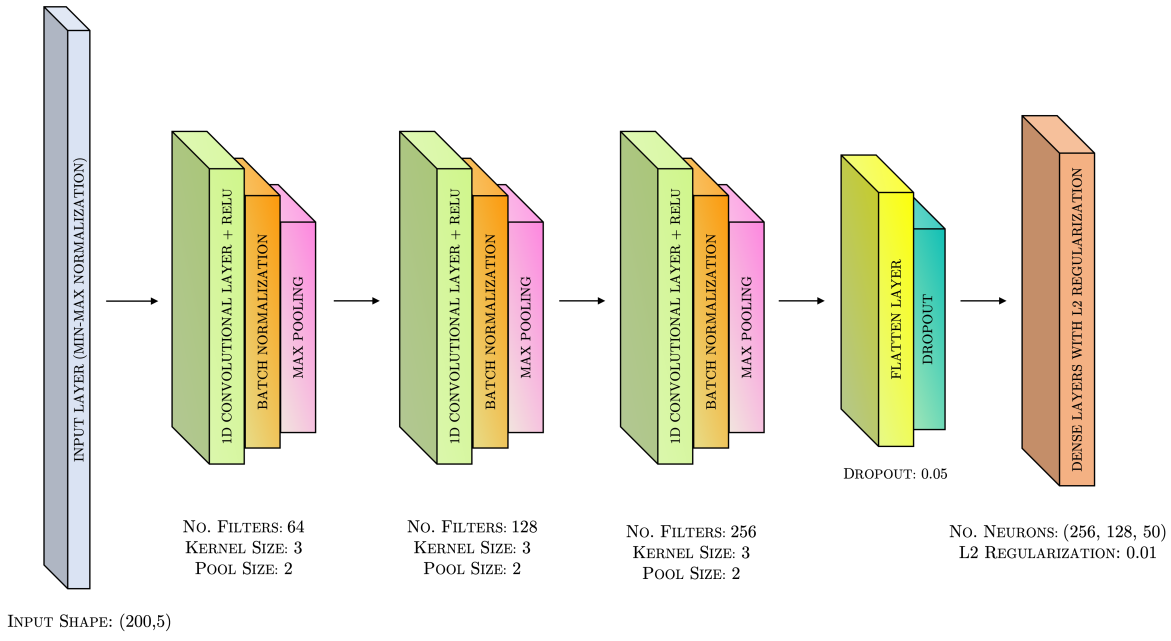


Figure 4.2: Architecture of the one-dimensional convolutional model.

Remark 1 *Batch normalization [57] is a technique that enables accelerating and stabilizing the training process of ANNs. The method involves normalizing (centering and scaling) each mini-batch that enters the network using the mean and standard deviation calculated from that specific mini-batch. The data is then re-scaled and re-centered using a set of parameters learned by the ANN. Moreover, since the mean and standard deviation are computed per mini-batch rather than the entire training set, batch normalization introduces some noise that serves as a regularization method and helps reduce overfitting.*

Hyperparameters and Learning Process

The model's hyperparameters have been fine-tuned to find a trade-off between variance and bias. The hyperparameters related to the architecture of the one-dimensional convolutional model are indicated in Figure 4.2, while those related to the training phase are reported in Table 4.1.

Hyperparameter	Value
Optimizer	Adam
Learning Rate	Starting from 10^{-2}, the learning rate was reduced by 3% each time there was no improvement in the loss function (evaluated over the training set) for 15 epochs. The minimum value it could reach was 10^{-5}.
Batch size	200 (all training observations)
Epochs	5000

Table 4.1: Learning hyperparameters of the one-dimensional convolutional model.

Remark 2 *The implemented variable learning rate technique gradually decreases the learning rate if there is no improvement in a given metric over a certain number of epochs (in this study, this metric is the RMSE evaluated on the training set). In other words, the learning rate remains unchanged when the reference metric improves but is reduced when the model's learning process stagnates. This is useful, for instance, to avoid oscillations around minimums. Another advantage of this method is that it saves time in manually optimizing this hyperparameter, as the learning rate is automatically self-regulated (exploring a wide set of values), depending on the evolution of the metric used as a reference.*

It is worth mentioning that the average execution time per epoch was approximately 50 ms when using the NVIDIA GPU, resulting in a total training time of 4 minutes. If the GPU was not used, the execution time per epoch was 250 ms, and the total training time exceeded 20 minutes.

Results of the One-Dimensional Convolutional Model

Regarding the results of the one-dimensional convolutional model, an RMSE of 4.95ºC was obtained in the training set and an RMSE of 16.26ºC in the test set. This indicates high variance or overfitting, despite regularization (dropout combined with L2-regularization). However, considering that the training set is small and that the measurement uncertainty of the outlet pyrometer fluctuates between 10ºC and 20ºC depending on the temperature, the result is not of such poor quality. For instance, Figure 4.3 illustrates some temperature profile predictions obtained with the one-dimensional convolutional model for both the training and test sets, showing that the model is capable of predicting the temperature profile quite accurately.

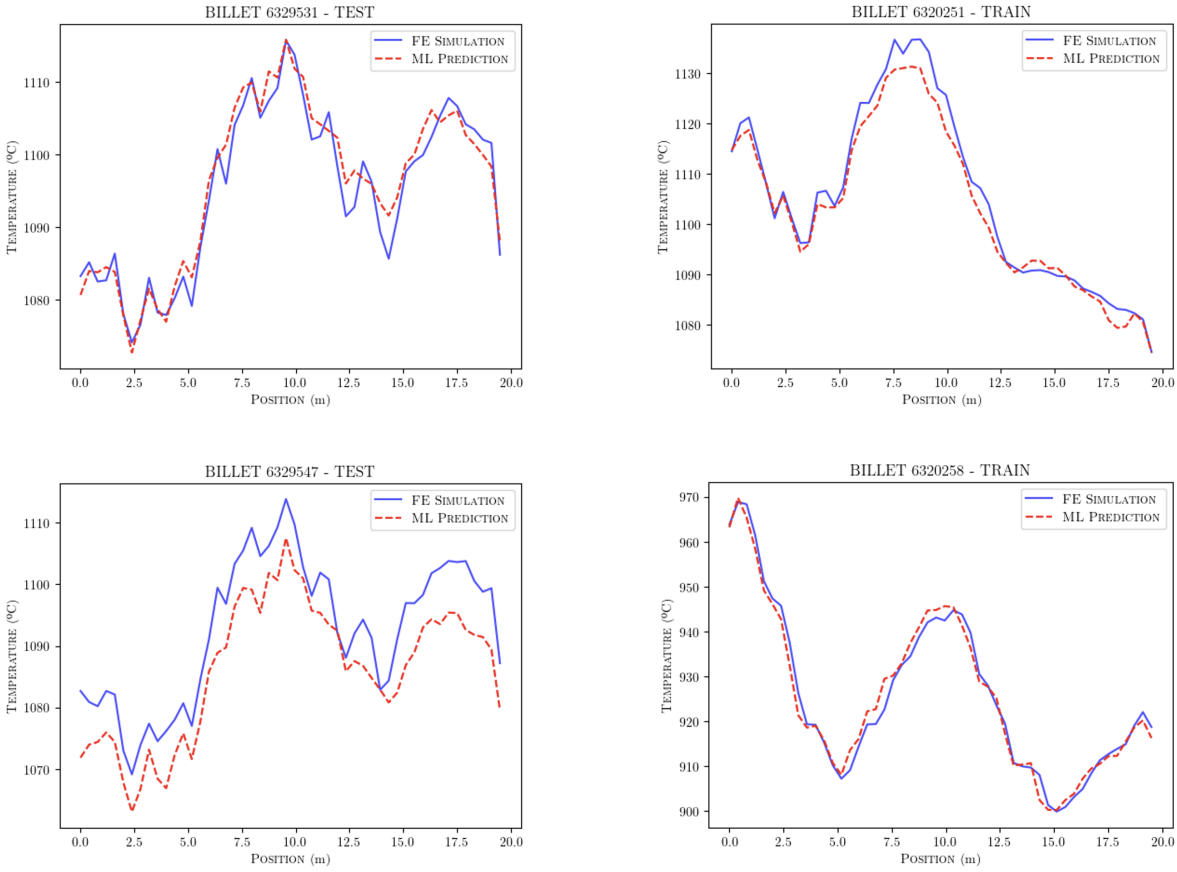


Figure 4.3: Comparison between the simulated temperature profiles using the FE model and those predicted using the one-dimensional convolutional model for two observations from the test set (first column) and two observations from the training set (second column).

4.2.2 Convolutional-Recurrent Model

This model is an adaptation for a regression problem of the ANN-based model proposed in the paper "*Multivariate LSTM-FCNs for Time Series Classification*" [58]. The model's architecture combines convolutional blocks with LSTM cells, as can be observed in Figure 4.4. One peculiarity of this architecture is that the convolutional blocks include an attention mechanism based on squeeze and excite blocks [59]. This mechanism assigns weights to each of the channels in the convolutional block, allowing the network to adaptively adjust them (they are trainable parameters) by giving more importance to some feature maps than others. However, what is perhaps most striking about the architecture is the dimension shuffle layer.

The dimension shuffle layer transposes the temporal dimension of the inputs to the convolutional blocks [58], which are multivariate time series consisting of 200 time steps with 5 different channels or variables (the advance of the billet and the power of each of the four groups of inductors) per time step. The reason for including this layer is the following: if the dimension shuffling is not applied, the LSTM would have to process 200 time steps with 5 variables each, whereas if the dimension shuffling is applied, the LSTM would have to process 5 time steps with 200 variables each. In other words, this dimension shuffling reduces the number of time steps the LSTM has to process from 200 to just 5, which is precisely the number of variables [58].

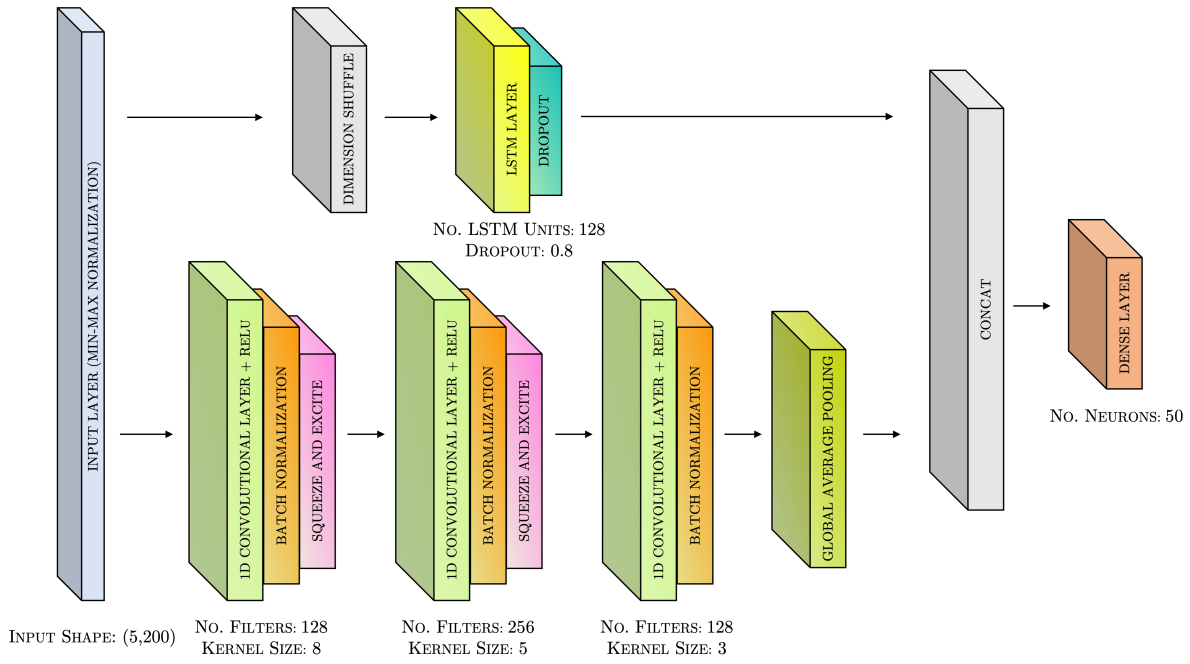


Figure 4.4: Architecture of the convolutional-recurrent model.

More formally, after the dimension shuffling, at each time step t, where $1 \leq t \leq 5$, the LSTM receives as input the univariate time series of length 200 that represents how variable t evolves with time [58]. Thus, the LSTM receives all the information about each of the variables together. This dimension shuffling technique helps reduce the training time, especially in cases where the number of variables is much smaller than the number of time steps (before the dimension shuffling), as in the problem studied. Even though it might seem that with this methodology there is information leakage (the correlation between variables is somehow lost), it has been proven in [58] that the performance obtained with this model is statistically the same as that of other models that do not use this dimension shuffling technique. Moreover, in [58], it has also been proven that the model with the architecture of Figure 4.4 outperforms other models that do not use dimension shuffling in a large number of UCR datasets [60].

Hyperparameters and Learning Process

Figure 4.4 indicates the hyperparameters associated with the convolutional-recurrent model's architecture, while Table 4.2 provides the hyperparameters specific to the training phase.

Hyperparameter	Value
Optimizer	Adam
Learning Rate	Starting from 10^{-2}, the learning rate was reduced by 2% each time there was no improvement in the loss function (evaluated over the training set) for 15 epochs. The minimum value it could reach was 10^{-5}.
Batch size	200 (all training observations)
Epochs	7000

Table 4.2: Learning hyperparameters of the convolutional-recurrent model.

In terms of execution time, each epoch took an average of 120 ms to complete when the NVIDIA GPU was used, resulting in a training time of about 14 minutes. However, if the GPU was not used, the execution time of an epoch shot up to 1s 700 ms, causing the total training time to reach almost 3.5 hours.

Results of the Convolutional-Recurrent Model

The convolutional-recurrent model achieves an RMSE of 5.70ºC on the training set and 9.85ºC on the test set. This indicates little degree of overfitting in the model, although to a lesser extent compared to the one-dimensional convolutional model. The fact that the error in the test set has now been reduced to 9.85ºC means that it is below the measurement error of the outlet pyrometer, which ranges between 10ºC and 20ºC, thereby enabling its usage within the RL environment (the error in the test set for the one-dimensional model was almost twice as high, exceeding the pyrometer uncertainty threshold). Figure 4.5 shows some predictions of the temperature profiles obtained with the convolutional-recurrent model for both the training and test sets.

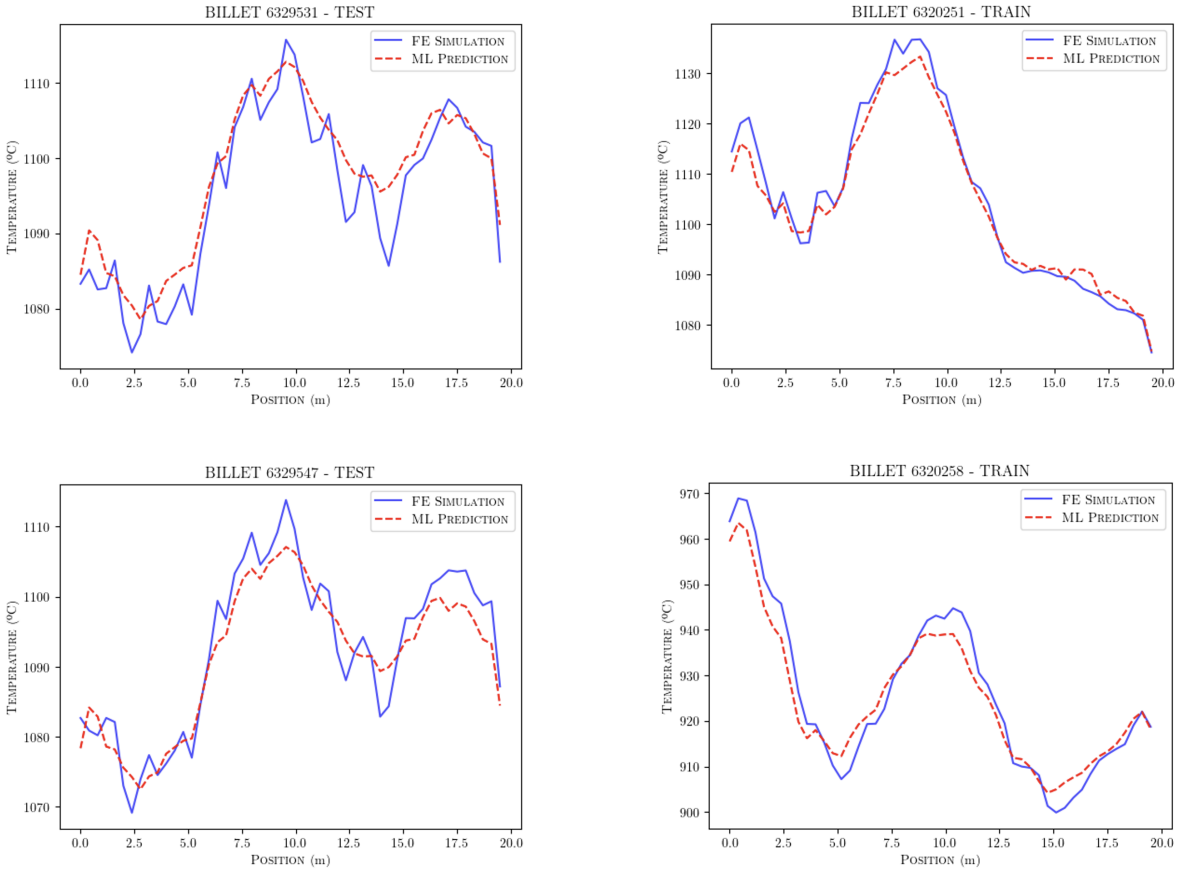


Figure 4.5: Comparison between the simulated temperature profiles using the FE model and those predicted using the convolutional-recurrent model for two observations from the test set (first column) and two observations from the training set (second column).

Chapter 5

Optimizing the Energy Consumption of the Induction Furnace

This chapter addresses the third phase of the project, which aims to introduce a proof-of-concept RL framework to optimize the energy consumption of GSW's induction furnace during the billets' heating process. Specifically, it outlines how a typical DQN-based RL scheme is tailored to the problem being studied by providing a description of its key components, such as the environment, the agent, the state space, the action space, and the reward system, among others. Furthermore, the chapter shows some preliminary results obtained for the resulfurized family of billets and discusses how to interpret them.

It should be noted that the ultimate purpose of this third phase is not to provide a definitive solution to the problem, but rather to serve as groundwork for the development of a RL-based framework to be enhanced in later phases of the project. Indeed, meaningful results cannot be delivered until GSW has fully defined the requirements and strategy, the FE model has been improved by incorporating the thermal losses of the inductors, and the ML model that plays the role of the virtual environment in the RL framework has been retrained with synthetic data. Thus, the goal of this chapter is to describe this framework, outline the workflow, and provide an example of how some preliminary results would be interpreted for a specific family of billets (namely, the resulfurized family). Hence, the results presented and analyzed in this chapter are neither binding nor global, but serve as a proof of concept of what could be achieved using the proposed methodology.

5.1 Configuration of the Reinforcement Learning Framework

When configuring a RL framework, it is necessary to determine who is going to play the role of each of its basic elements in the problem being studied. The first step involves identifying the agent and the environment. In this research, a DQL approach has been employed to tackle the RL problem, and therefore, the agent is represented by a DQN, as mentioned in Section 2.3. On the other hand, the environment with which the agent interacts is a virtual prototype of GSW's real induction furnace, and it is emulated by the ML model that combines convolutional and recurrent neural networks (see Section 4.2.2). This ML model is preferred over the one based exclusively on CNNs because of its higher generalization capability (the one-dimensional convolutional model had a higher variance). Figure 5.1 shows the structure of the proposed RL

framework, particularized to the problem under study.

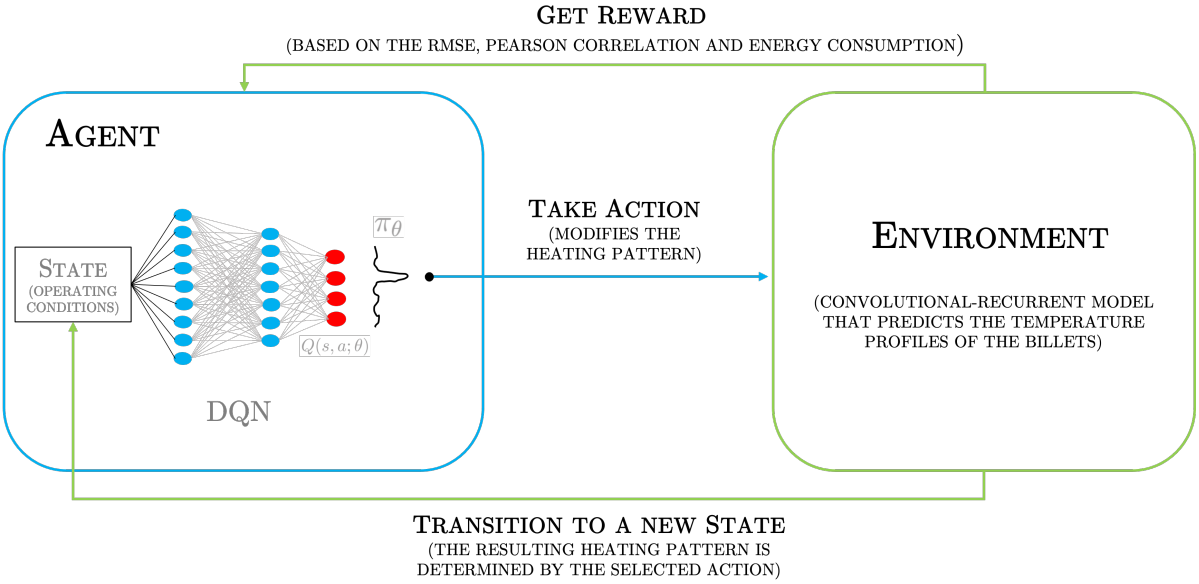


Figure 5.1: Structure of the proposed RL framework particularized to the problem under study.

Once the agent and the environment have been established, the next step is to determine the objective to be pursued by the agent when making decisions (e.g., in the context of chess, the goal would be to achieve a checkmate against the opponent's king).

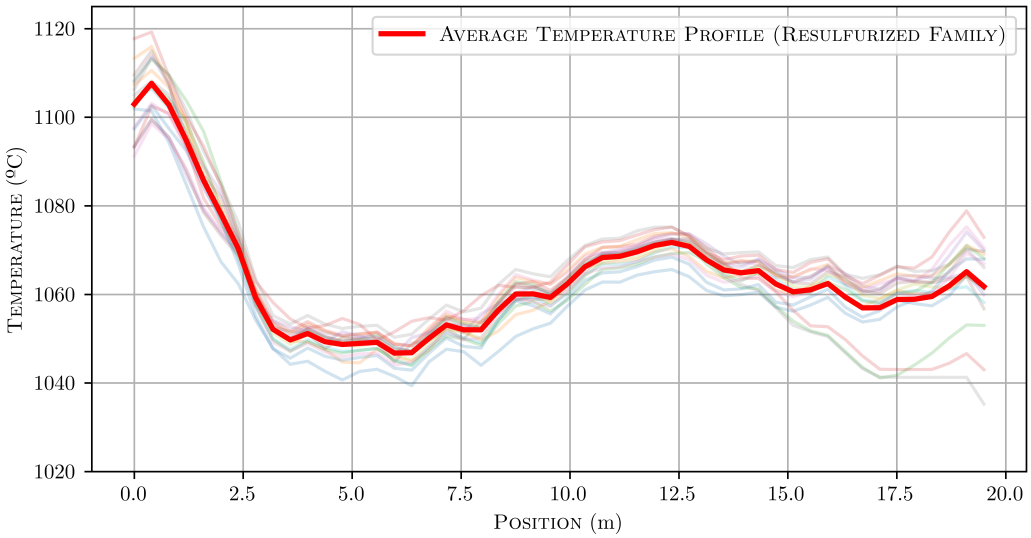


Figure 5.2: Target temperature profile of the DQN agent. The shaded curves correspond to the (simulated) temperature profiles associated with the billets belonging to the resulfurized family, while the thick red curve represents the average temperature profile of the resulfurized billets.

In the absence of clear criteria defined by GSW, the target for the DQN agent is set to discover states (or operating conditions, see Section 5.1.1) whose associated temperature profiles are as close as possible to the average temperature profile of the billets belonging to the resulfurized family. This approach, which is quite conservative as too hot or too cold temperature profiles are avoided, guarantees that there is no prominent bias. Figure 5.2 shows the target temperature profile along with the simulated temperature profiles of the resulfurized billets.

Remark 3 *It is worth highlighting that the temperature profile set as the agent's target is indicative, meaning that the temperature profiles associated with the new states found by the agent are not required to exactly match it. Indeed, a temperature profile with minor upward or downward temperature variations remains equally valid as long as it is similar to the target temperature profile (two temperature profiles are considered similar if their RMSE is below $5^{o}C$, as will be further discussed in Section 5.1.3). Thus, there is a similarity margin encompassing all temperature profiles whose RMSE with respect to the target temperature profile is below $5^{o}C$, which can be used by the agent to look for operating conditions that reduce energy consumption.*

In addition to the agent, the environment, and the agent's target, there are still some components that have to be defined within a RL framework, namely: the state space, the action space, and the reward system. These aspects are discussed in the following subsections.

5.1.1 State Space

Each state vector, which represents both the advance of the billet inside the induction furnace and the heating pattern (the four time series accounting for the instantaneous powers of the four inductor groups) applied to it, consists of 1000 values arranged in a 200×5 matrix as follows:

$$s_t = \begin{pmatrix} x_1 & x_2 & \dots & x_{199} & x_{200} \\ p_{1,1} & p_{1,2} & \dots & p_{1,199} & p_{1,200} \\ p_{2,1} & p_{2,2} & \dots & p_{2,199} & p_{2,200} \\ p_{3,1} & p_{3,2} & \dots & p_{3,199} & p_{3,200} \\ p_{4,1} & p_{4,2} & \dots & p_{4,199} & p_{4,200} \end{pmatrix}^T,$$

where $x_i \geq 0$, with $\sum_{i=1}^{200} x_i = 40$ (twice the furnace's length), denotes the billet's advance in the i-th simulation step, measured in meters. Similarly, $p_{j,i} \in [0, 1000]$ denotes the instantaneous power of the j-th group of inductors in the i-th simulation step, with $j \in \{1, 2, 3, 4\}$, and is measured in kilowatts. The simulation has been configured to have 200 one-second steps, so $i \in \{1, \dots, 200\}$. Moreover, the upper and lower bounds for the power have been agreed upon with GSW.

Initial State

It should be noted that RL is an iterative process that requires an initial state from which the agent can begin taking actions. Ideally, this initial state would be the null state (with all its entries set to zero), so that the agent builds the optimal state based on the experience it acquires. However, this presents two challenges, at least in this preliminary proof-of-concept: one computational and another operational.

On the one hand, the computational challenge is directly related to the continuous nature of the space state. Indeed, the fact that the state space is continuous and not discrete implies that, if starting from the null state, the agent would require numerous iterations and a very

fine granularity in the action space to be able to simulate a feasible and realistic state similar to that of Figure 2.1 (excluding the pyrometer recordings, as they are not part of the operating conditions). If reproducing a single state is already challenging, exploring the entire state space becomes impractical without taking measures to reduce the computational cost.

On the other hand, the operational problem is that, if the agent is left to learn from scratch, it may learn operating conditions that cannot be applied to billets in GSW's furnace, either because they require higher working powers or because they are operationally unfeasible, for instance, requiring equipment that can adjust the powers in a faster and more local way.

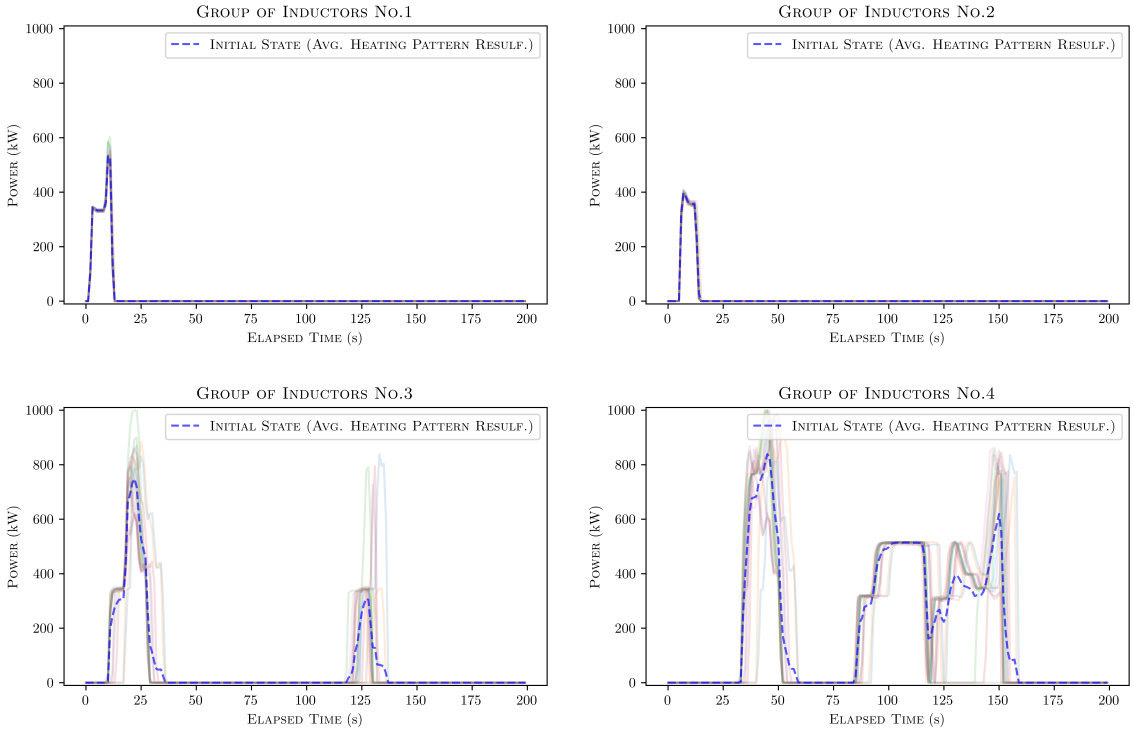


Figure 5.3: Heating pattern associated with the initial state (thick dashed blue line), which serves as the starting point for taking actions in the RL model. It corresponds to the average heating pattern applied by GSW to the resulfurized billet family, shown as shaded curves.

Therefore, in the absence once again of a definite strategy, the average of the operating conditions currently applied by GSW when heating the resulfurized billets has been chosen to be the initial state from which the agent can begin taking actions and interacting with the virtual environment. The main reason to follow this criterion is that, although it limits the optimization process by restricting it to a very specific area of the state space (in some way, it starts from a very specific state, characteristic of the resulfurized family), it guarantees that the operating conditions discovered by the agent are feasible in GSW's furnace or at least are similar to current practices (setting upper limits for the instantaneous power, of course). Figure 5.3 shows the heating pattern associated with this initial state (the advance of the billet inside the furnace is not shown).

Based on the foregoing, it is worth highlighting that the ultimate goal is not to look for the pattern that optimizes the overall billet heating process, but rather to use the RL framework to find actions that lead to new operating conditions that reduce energy consumption. In the particular case of the illustrative example at hand, the aim is to use the proposed RL-based methodology to enhance the operating conditions that GSW currently uses for heating billets of the resulfurized family, seeking how to modify them to reduce energy consumption, rather than finding a global optimal pattern for that family (at least with the current state of the research).

5.1.2 Action Space

One of the challenges of the problem being studied is that the state space is continuous (e.g., the instantaneous power of the inductors can take any value between 0 and 1000 kW). Consequently, the space of possible actions is also continuous. For instance, an action could consist of increasing the power of a group of inductors at a given instant by 10 kW, 10.1 kW, 10.11 kW, 10.111 kW, and so on, with the theoretical possibility of infinitely making small increments, even though these actions would be indistinguishable in real life. Moreover, each of these actions can be taken instantaneously at each of the 200 steps of the simulation, unveiling the complexity of working with a continuous action space.

In order to reduce the computational cost and develop a feasible RL model, the action space has been discretized. Furthermore, in this preliminary proof of concept, the billet's advance pattern inside the furnace has been kept intact, so that the actions are applied exclusively to the heating patterns (time series of the instantaneous power of the four inductor groups). Specifically, the actions have been considered to have a granularity of 20 simulation steps, i.e., instead of being applied instantaneously, they are applied as a whole in batches of 20 contiguous steps.

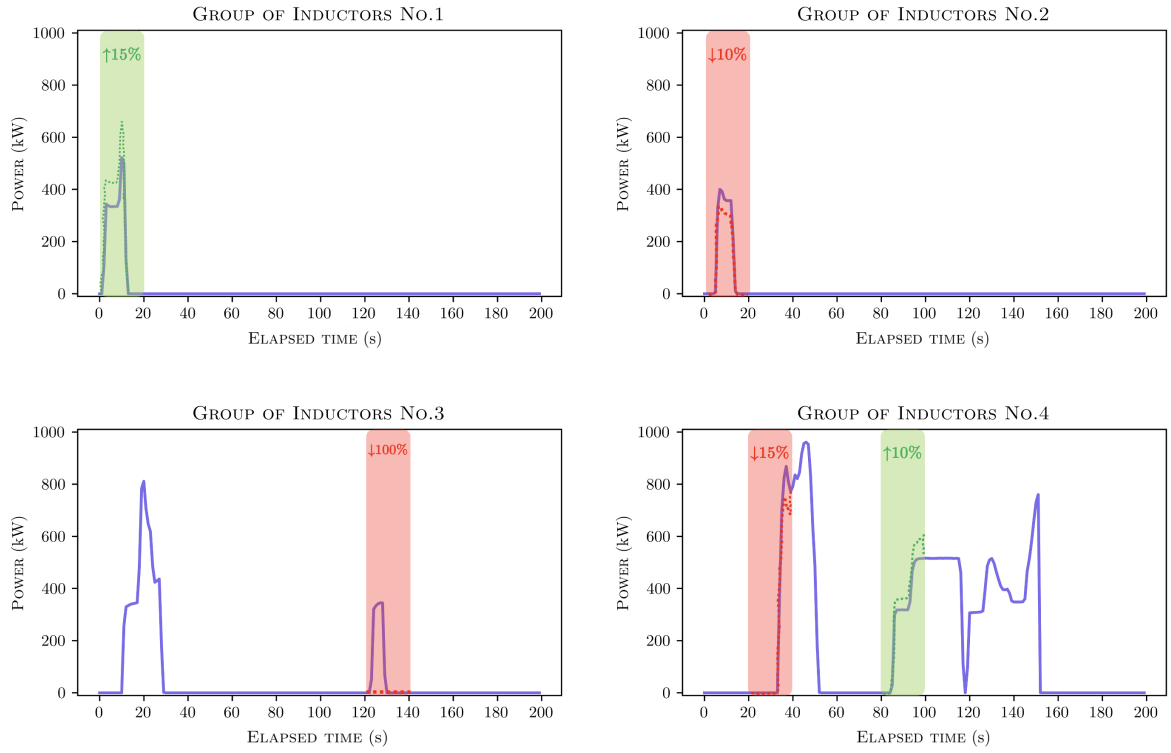


Figure 5.4: Some discretized actions applied on a billet heating pattern.

It has been initially agreed with GSW that the agent can adjust the power of the inductor groups separately in batches of 20 steps by means of 5 operations: increasing or decreasing the power by 10%, increasing or decreasing the power by 15%, and decreasing the power by 100% (partially turning off the inductors). Considering that each time series consists of 200 steps and actions are used to modify four power time series separately, the cardinality of the discretized action space is (200/20) batches $\times$ 5 operations $\times$ 4 time series = 200 actions. Figure 5.4 shows some examples of how these discretized actions that are available to the agent could be applied.

For instance, if the agent were to select an action that involves increasing the power of the second group of inductors by 15% for the initial 20 steps, the environment's current state, s_t, would transition towards a new state, s_{t+1}, which would be expressed as follows:

$$s_{t+1} = \begin{pmatrix} x_1 & x_2 & \dots & x_{20} & x_{21} & \dots & x_{199} & x_{200} \\ p_{1,1} & p_{1,2} & \dots & p_{1,20} & p_{1,21} & \dots & p_{1,199} & p_{1,200} \\ 1.15 \cdot p_{2,1} & 1.15 \cdot p_{2,2} & \dots & 1.15 \cdot p_{2,20} & p_{2,21} & \dots & p_{2,199} & p_{2,200} \\ p_{3,1} & p_{3,2} & \dots & p_{3,20} & p_{3,21} & \dots & p_{3,199} & p_{3,200} \\ p_{4,1} & p_{4,2} & \dots & p_{4,20} & p_{4,21} & \dots & p_{4,199} & p_{4,200} \end{pmatrix}^T .$$

5.1.3 Reward System

Once the agent has selected an action, the current state transitions to a new state, and then the environment (the ML model) simulates the temperature profile associated with it. Right after that, a reward is computed, which is later used to update the agent's decision-making policy and evaluate whether the selected action helps the agent move in the right direction and achieve its goal. The reward in the problem under study has been defined so that it depends on both the RMSE and the Pearson correlation coefficient between the temperature profile simulated by the ML model and the target temperature profile, as well as the energy consumption. Formally, the implemented reward function is expressed as follows:

$$r_t = \begin{cases} -10^6 - \text{RMSE}(T^{\text{obj.}}, T^{s_t}), & \text{if RMSE}(T^{\text{obj.}}, T^{s_t}) \geq 25. \\ -10^4 - \text{RMSE}(T^{\text{obj.}}, T^{s_t}), & \text{if RMSE}(T^{\text{obj.}}, T^{s_t}) \in [10, 25). \\ -10^3 - \text{RMSE}(T^{\text{obj.}}, T^{s_t}), & \text{if RMSE}(T^{\text{obj.}}, T^{s_t}) \in [7, 10). \\ -10^2 - \text{RMSE}(T^{\text{obj.}}, T^{s_t}), & \text{if RMSE}(T^{\text{obj.}}, T^{s_t}) \in [5, 7). \\ (100 - \text{E}^{s_t}) \cdot \rho(T^{\text{obj.}}, T^{s_t}), & \text{if RMSE}(T^{\text{obj.}}, T^{s_t}) < 5, \end{cases}$$

where $\text{RMSE}(T^{\text{obj.}}, T^{s_t})$ and $\rho(T^{\text{obj.}}, T^{s_t})$ are, respectively, the root mean squared error and the Pearson correlation coefficient between the target temperature profile, $T^{\text{obj.}}$, and the temperature profile simulated by the ML model (the RL environment) for the state s_t, T^{s_t}.

Remark 4 *The energy consumption, measured in kilowatt-hours (kWh), has been calculated as the sum of the area under the curve of the four time series accounting for the instantaneous power of the four groups of inductors. Following the notation introduced in Section 5.1.1, the energy consumption computation can be expressed as follows:*

$$\text{E}^{s_t} := \frac{1}{3600} \sum_{j=1}^{4} \sum_{i=1}^{200} p_{j,i} \cdot \Delta t_i,$$

where $\Delta t_i = 1s$, $\forall i \in \{1, \dots, 200\}$, *and* $\frac{1}{3600}$ *is a conversion factor from kW-second to kW-hour.*

From the expression above, it follows that the reward increases as the RMSE decreases, so that the highest rewards are assigned when it can be ensured that the simulated temperature profile is very similar to the target profile. Specifically, the criterion initially agreed upon with GSW is that two temperature profiles are very similar if the RMSE between them is less than 5ºC (it is recalled that the pyrometer measurement error ranges between 10ºC and 20ºC, depending on the temperature). Thus, the implemented reward system especially penalizes operating conditions or states that result in temperature profiles with an RMSE greater than 5ºC, where the penalty decreases as the threshold of an RMSE of 5ºC is approached.

This definition of the reward function encourages the agent to first focus on finding states that result in temperature profiles very similar to the target. Only after this similarity criterion is satisfied does the agent focus its efforts on reducing energy consumption. Indeed, when the RMSE is less than 5ºC, the reward no longer depends on the RMSE, but becomes dependent on both the energy consumption and the Pearson correlation coefficient between the target and simulated temperature profiles. In particular, the reward becomes positive only when the correlation coefficient is positive. Furthermore, it reaches its maximum value when, in addition to the correlation being equal to 1, energy consumption is minimal. The correlation is included as a correction factor to reward those temperature profiles that not only fulfill the RMSE criterion, but also exhibit a strong positive correlation with the target temperature profile.

5.1.4 Deep Q-Network Architecture and Training

Once all the components of the RL framework have been defined, the next step is to train the DQN representing the agent. Specifically, the implemented DQN is an FCNN that takes the state or operating conditions as inputs, and outputs the Q-values associated with each of the 200 possible actions within the action space. Consequently, the output layer, which has a linear activation function, is formed by 200 neurons, each one representing a specific action. In addition, the DQN includes four dense hidden layers with ReLU activation and He-weight initialization [61] that are composed, respectively, by 1024, 1024, 512, and 256 neurons.

Regarding the training phase, the MSE has been used as the loss function, and *Adam* has been used as the optimizer. Moreover, a learning rate (step size of the gradient descent algorithm) of 0.01 and a batch size of 64 have been used. The learning process has been executed for 500 episodes, each of which involves a maximum of 100 weight updates if the stopping criterion is not met. In particular, the stopping criterion has been set to achieve a positive reward and an energy consumption below 10 kWh, which is slightly lower than the minimum energy consumption observed among the resulfurized family. Therefore, the DQN weights have been dynamically updated up to a total of 50000 times. The average execution time per episode using the NVIDIA GPU is approximately 9 seconds, resulting in a total training time of 1 hour and 15 minutes.

5.2 Analysis of Reinforcement Learning-Driven Results

Once the DQN has been trained, the outcome is a decision-making policy (see Section 2.3). The procedure to take advantage of this policy in order to identify heating patterns that reduce energy consumption is as follows. Firstly, a steel billet is selected, and the current operating conditions used by GSW in the factory for heating that specific billet are set as the initial state. Right after that, the learnt decision-making policy is iteratively used to select actions following an epsilon-greedy criterion (see Equation 2.12) and modify this initial state. Ideally,

this epsilon-greedy criterion should assign a low probability to randomly exploring the action space to encourage the agent to exploit the learnt decision-making policy. This iterative process is executed for a certain number of episodes, where the number of actions selected in each of them varies based on whether the stopping criterion is met or not.

If everything goes according to plan, the outcome of this iterative process is potentially a new state or set of operating conditions that could exhibit reduced energy consumption, with its associated temperature profile being very similar to the target profile. The optimized state corresponds to the state that has achieved the highest reward throughout the iterative process.

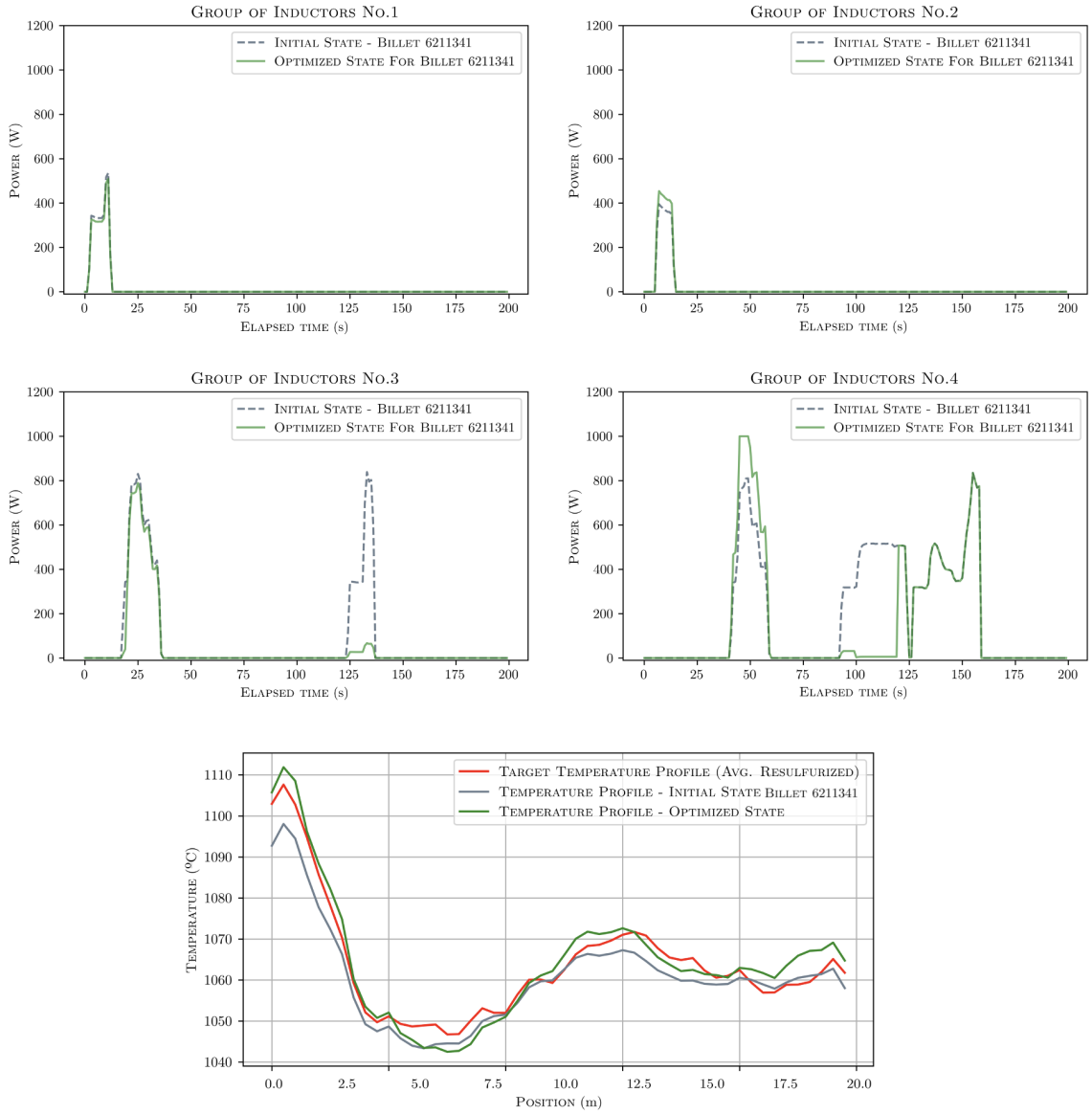


Figure 5.5: Results of energy consumption optimization for billet with ID 6211341 in the resulfurized family. The heating patterns are shown in the first two rows, where the gray curve represents the currently used pattern, and the green curve represents the optimized one obtained using the proposed RL pipeline. In the last row, the target temperature profile (red curve) is compared with the current (gray curve) and optimized (green curve) temperature profiles.

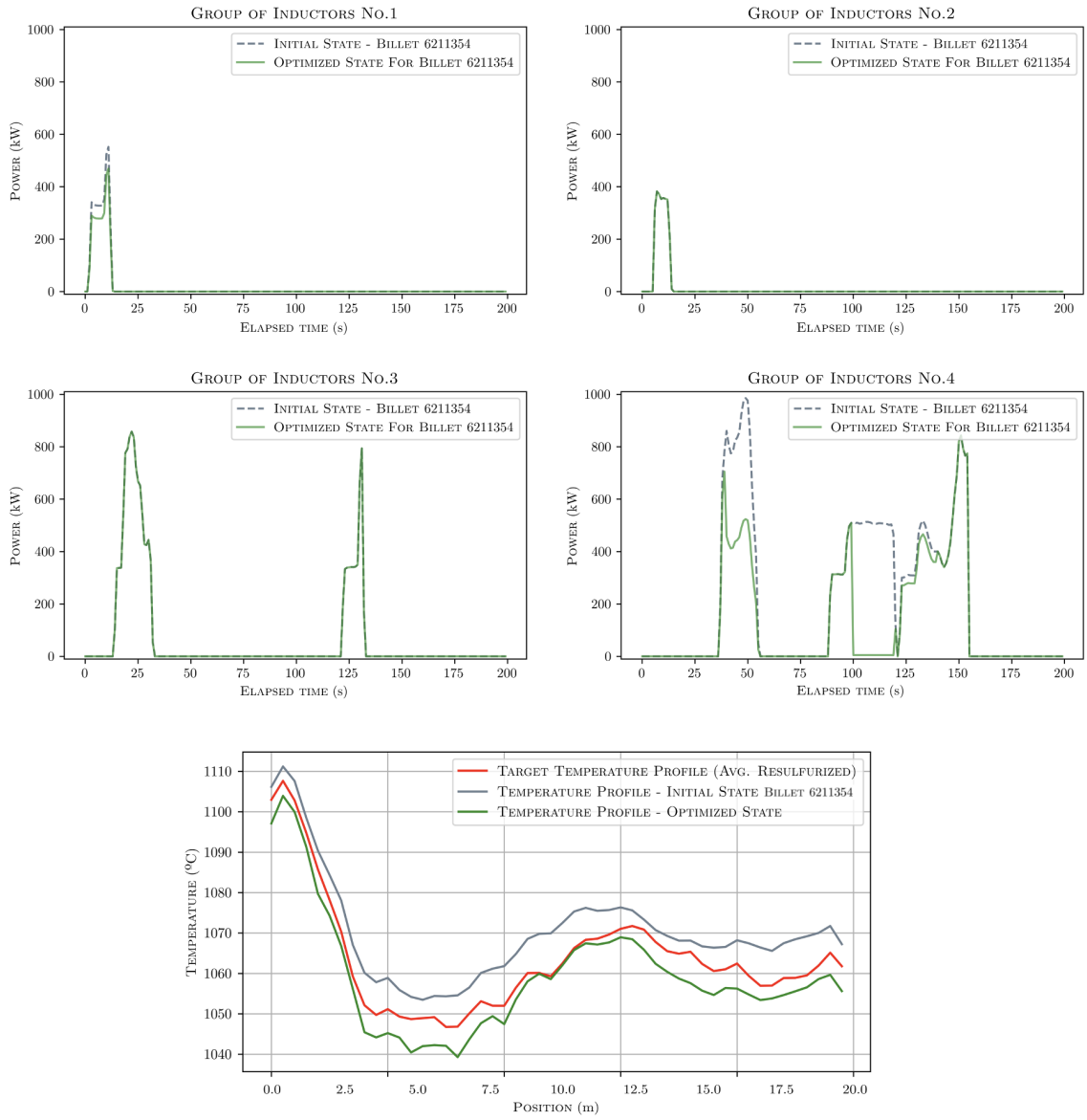


Figure 5.6: Results of energy consumption optimization for billet 6211354 in the resulfurized family. The arrangement and meaning of the different curves are the same as in Figure 5.5.

The aforementioned RL pipeline has been tested on the operating conditions related to the heating of the billets belonging to the resulfurized family, as indicated in this chapter's introduction. In particular, a total of 300 episodes have been executed in the iterative process. In each of them, the initial state is reset, and a maximum of 1000 actions are sequentially selected based on the learnt policy (they are less if the stopping criterion is met, which, as in the training phase, consists of obtaining a positive reward and an energy consumption below 10 kWh). Furthermore, a probability of 0.2 has been set for random exploration, which means that in 80% of the cases the action is chosen based on the learnt policy and only in 20% of the cases it is chosen at random. It is worth mentioning that the reason to choose a non-zero probability of random exploration is that it allows for better coverage of the action space and prevents stagnation. Figure 5.5 and Figure 5.6 show the results obtained when applying the

proposed methodology to the operating conditions of two billets belonging to the resulfurized family, serving as illustrative examples.

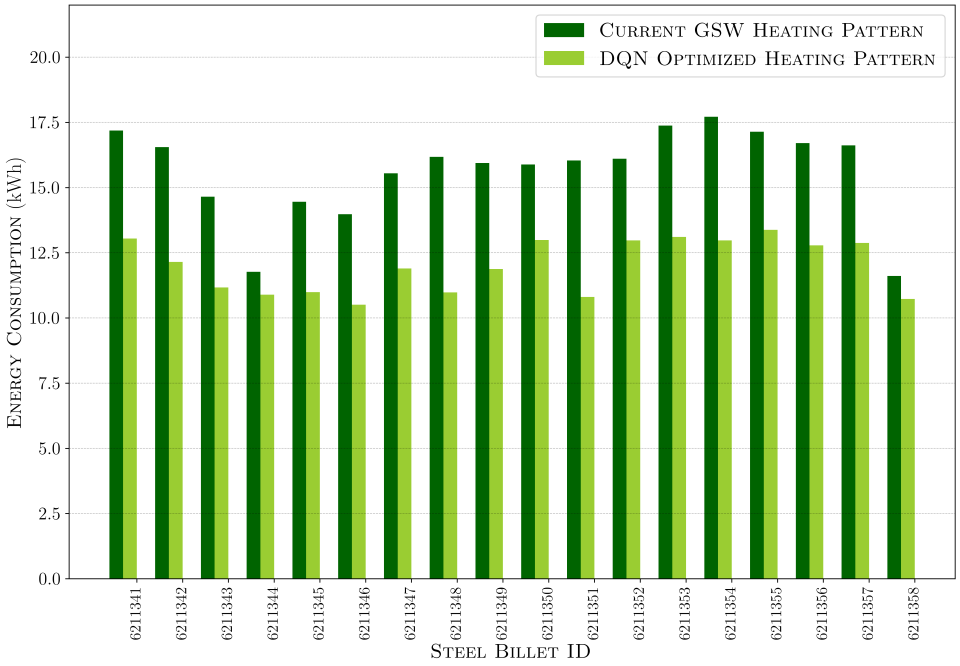
On the one hand, Figure 5.5 illustrates a situation where the operating conditions currently applied by GSW (gray curves) result in a temperature profile that is "colder" than the target profile. Consequently, in this case, the challenge for the RL agent is to look for a state with an associated temperature profile as hot as the target profile, while reducing energy consumption if possible. It is observed that the RL agent has indeed been able to find a new state (green curves) with an associated temperature profile that is quite similar to the target profile (quantitatively, the RMSE is 3.48^{o}C). Moreover, this new state is associated with an energy consumption of 13.05 kWh compared to the 17.2 kWh of the initial state, indicating energy savings.

Specifically, the green curves in the first two rows of Figure 5.5 show one way in which the heating pattern currently used by GSW to heat the billet with ID 6211341 could be modified to achieve the desired temperature profile in a more energy-efficient manner. For instance, the plots suggest that this can be potentially achieved by slightly increasing the power of the second and fourth groups of inductors during the first 75 seconds, while delaying the second activation of the fourth group of inductors, and almost completely deactivating the third group of inductors once the billet has been inside the furnace for 50 seconds.

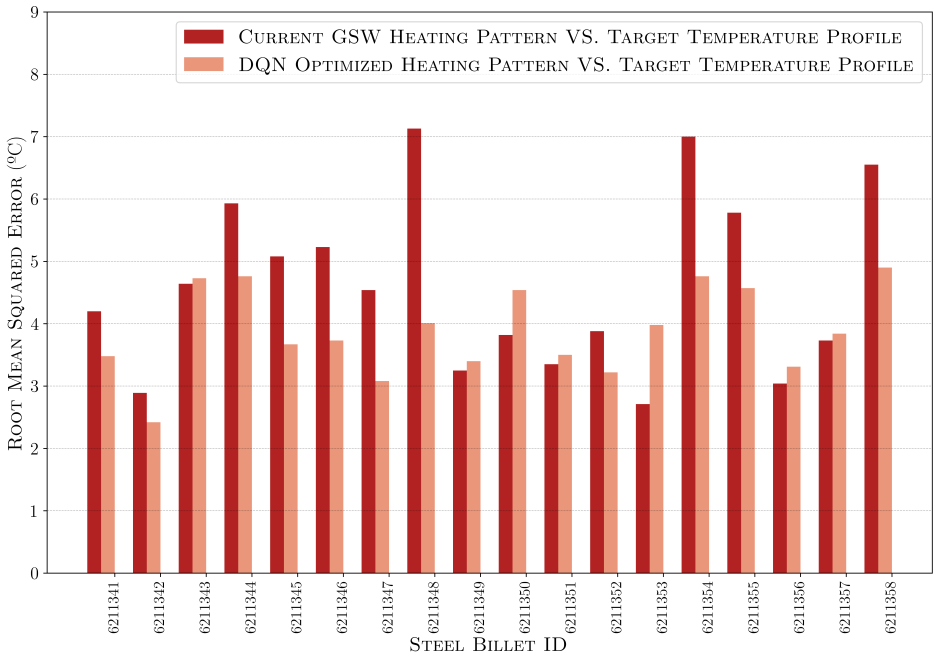
On the other hand, Figure 5.6 shows the opposite situation, i.e., when the initial operating conditions lead to a temperature profile that is "hotter" than the target temperature profile, suggesting that the heating pattern currently used by GSW to heat the billet with ID 6211354 uses too high powers in some inductor groups, presumably consuming more energy than necessary. Therefore, it is reasonable to expect that the RL agent is going to be able to find new operating conditions with lower powers that allow reaching the target temperature profile while consuming less energy.

The heating pattern identified by the RL agent, represented by the green curves in the first two rows of Figure 5.6, suggests that it is not necessary to apply as much power to the fourth group of inductors to reach the desired temperature profile. In fact, the plots indicate that simply reducing the power of the fourth inductor group to almost half in the first 75 seconds and temporally turning off its inductors for approximately 25 seconds prior to the third activation not only still allows to obtain a profile similar to the target (the RMSE between the target profile and that of the new state is 4.76°C), but also allows to do so while consuming less energy. Specifically, the consumption of this new state is 12.97 kWh, whereas the consumption associated with the initial state was 17.72 kWh.

Thus far, the results for two billets from the resulfurized family have been analyzed and discussed as examples to illustrate how the results obtained with the proposed RL pipeline would be interpreted. To complete this information, Figure 5.7 presents a summary of the results obtained for all 18 billets within the resulfurized family. Figure 5.7a shows the energy consumption associated with the operating conditions currently used by GSW to heat each billet (dark green), as well as the energy consumption associated with the optimized states discovered through the RL pipeline (light green). Similarly, Figure 5.7b shows the RMSE between the temperature profile associated with the operating conditions employed by GSW to heat each billet and the target temperature profile (dark red), along with the RMSE between the target temperature profile and the temperature profile associated with the optimized states (light red).



(a) Comparison of energy consumption between current and optimized states.



(b) Comparison of RMSE between current and optimized temperature profiles.

Figure 5.7: Results of energy consumption optimization obtained using the proposed RL pipeline for the 18 billets that make up the resulfurized family.

It is observed in Figure 5.7a that the RL agent successfully identifies operating conditions that reduce energy consumption compared to the conditions currently used at GSW for all billets in the resulfurized family. Furthermore, Figure 5.7b shows that the RMSE between the target

temperature profile and the temperature profiles associated with these optimized states is always below the similarity threshold of 5ºC. This means that the states found by the agent not only result in lower energy consumption, but also generate temperature profiles very similar to the desired one.

However, it is seen for some billets that the RMSE corresponding to the most energetically favorable states found by the RL agent is higher than the RMSE obtained when considering the initial states. This means that, in certain cases, the reduction of energy consumption comes at the expense of the similarity between the target temperature profile and the one associated with the optimized state. Hence, although the optimized profiles still meet the established similarity criterion, the RL agent should ideally seek new operating conditions that reduce energy without worsening the RMSE associated with the initial state.

The latter requires caution when quantifying the reduction in energy consumption, since part of this reduction could have been achieved at the expense of a worsening in the RMSE towards a "colder" temperature profile (which presumably requires less power and, therefore, less energy), even within the established similarity margin. However, while these energy consumption quantitative values may not be entirely meaningful, they do suggest that the proposed methodology allows for finding energy-efficient operating conditions, even though the reduction in energy consumption may not be as high as reported in Figure 5.7a. Indeed, the impact of a worsening in the RMSE is not expected to be significant enough so that the agent would be suddenly unable to find operating conditions that cut down on energy consumption.

In short, despite the assumptions made, the lack of a fully defined strategy established by GSW, and the limitations in the performance of the FE and ML models, it can be concluded that the proposed RL pipeline allows for finding more energy-efficient billet heating operating conditions than those currently used by GSW, while achieving temperature profiles at the induction furnace's exit very similar to the target one (at least for the resulfurized family). However, it is expected that in future phases of the project, when the strategy and similarity criteria are better defined and more data is available to enhance the ML model (RL environment), similar results could be obtained for the remaining billet families.

Chapter 6

Conclusions and Future Work

This master's thesis applies AI to address one of the greatest challenges that the industrial sector has faced in recent years: the energy crisis. Specifically, this research focuses on one of the most complex metallurgical processes, the wire rod steel rolling process, which results in a material widely used in cable manufacturing and construction. More precisely, the project deals with the heating phase within this process, which is a really energy-consuming stage where steel billets (semi-finished steel products) undergo certain heating treatments in an induction furnace to enable their subsequent deformation and shaping.

In this master's thesis, a proof of concept is conducted to assess a methodology that combines FE numerical modeling, ML techniques, and RL in order to optimize the energy consumption of an induction furnace located at Global Steel Wire (GSW), a steelmaker situated in Santander (Cantabria, Spain). The details and conclusions related to each of the three major phases into which the project is divided are outlined in what follows.

In the first phase (addressed in Chapter 3), a FE digital twin of GSW's induction furnace has been developed that, although simplifying some physical processes, is capable of faithfully simulating the heating process of the billets. Specifically, the model has been calibrated for the resulfurized billet family, achieving RMSEs below 5^{o}C and Pearson correlation coefficients exceeding 0.95 between the simulated temperature profiles and those recorded by GSW's outlet pyrometer. The main role of this FE model in the proposed workflow has been the generation of background data, given its robustness and extrapolation capabilities.

In the second phase (discussed in Chapter 4), two ML models based on ANNs have been implemented that are able to predict the temperature profiles of the billets at the furnace's exit in real time, leveraging the outcomes of the FE model developed in the previous phase. In particular, a model combining one-dimensional CNNs and LSTMs has been implemented, which yields an error slightly lower than 10^{o}C on the test set (below the measurement uncertainty of the outlet pyrometer). The main role of this convolutional-recurrent ML model has been to serve as a virtual environment for the induction furnace within a RL framework, something that would be unfeasible for the FE model due to its long simulation times.

Finally, in the third phase (covered in Chapter 5), a DQL framework has been introduced to identify billet heating patterns that reduce the energy consumption of GSW's induction furnace. Specifically, the agent is a DQN, and the virtual environment is the convolutional-recurrent ML model developed in the second phase. A preliminary proof of concept has been conducted to test

this pipeline using the resulfurized billet family. The results show that the methodology allows for finding heating patterns that reduce energy consumption while maintaining a temperature profile similar to the target one, although it is worth noting that the quantitative interpretability of the results is limited by the absence of a clear strategy and quality criteria.

Nevertheless, the proposed methodology has some limitations that would be interesting to address in future work, for instance:

- The FE model does not take into account the thermal losses of the inductors. However, there is already a group of project members working to develop and implement a theoretical electromagnetic model in the digital twin that incorporates these losses.
- The FE model uses the average temperature recorded by the inlet pyrometer as a boundary condition. Thus, it is possible to enhance the FE model by including the inlet temperature time series as an additional input. This modification is expected to be useful to reproduce the cold spots shown in Figure 3.3.
- The dataset provided by GSW (consisting of the heating operating conditions for 217 billets) is insufficient to train the ML models. It would be interesting to expand the training set by generating synthetic samples using the FE model.
- ML models are somewhat overfitted, especially the model based on one-dimensional CNNs. To mitigate this, two potential solutions could be: (1) expanding the training set, as suggested in the previous item; and (2) exploring alternative ML models that manage to reduce variance.
- The strategy for applying the RL pipeline is not fully defined. Indeed, it would be necessary for GSW to provide further guidelines to, for instance: study how to define both the initial state and the target temperature profile to avoid biases, look for an appropriate granularity of the action space that is feasible to apply in the factory, or refine the similarity criterion between temperature profiles to enhance the interpretability of the quantitative results.

In spite of these limitations, it can be concluded that the results obtained in this preliminary proof of concept are promising and demonstrate the usefulness of the methodology proposed in this master's thesis to reduce energy consumption in the billets' heating process within GSW's induction furnace. Furthermore, it is expected that in the not-so-distant future, the introduced workflow can be further enhanced by incorporating the improvements indicated above, which would presumably lead to more significant results that could enable the implementation of this framework in GSW's real-life factory. Moreover, this proof of concept paves the way for the application of the proposed methodology in other industrial processes with similar conditions, such as glass manufacturing, metal casting, or petroleum refining, among others.

References

[1] Bruno Urmersbach. Europäische Union: Anteile der Wirtschaftssektoren am Bruttoinlandsprodukt (BIP) von 2011 bis 2021, 2022. Available online at: URL (accessed 24 May, 2023).

[2] Eurostat. Industry Relied Mostly on Natural Gas and Electricity, 2023. Available online at: URL (accessed 24 May, 2023).

[3] United Nations. 2030 Agenda - Sustainable Development Goals, 2015. Available online at: URL (accessed 24 May, 2023).

[4] United Nations. Paris Agreement, 2015. Available online at: URL (accessed 24 May, 2023).

[5] European Commission. European Green Deal, 2020. Available online at: URL (accessed 24 May, 2023).

[6] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. *Business and information systems engineering*, 6:239–242, 2014.

[7] Ricardo Silva Peres, Xiaodong Jia, Jay Lee, Keyi Sun, and Armando Walter Colombo. Industrial Artificial Intelligence in Industry 4.0 - Systematic Review, Challenges and Outlook. *IEEE Access*, 8:220121–220139, 2020.

[8] Jay Lee, Hossein Davari, Jaskaran Singh, and Vibhor Pandhare. Industrial Artificial Intelligence for Industry 4.0-Based Manufacturing Systems. *Manufacturing letters*, 18:20–23, 2018.

[9] Saurabh Vaidya, Prashant Ambad, and Santosh Bhosle. Industry 4.0 - A Glimpse. *Procedia manufacturing*, 20:233–238, 2018.

[10] Filipe Pereira, Vítor Carvalho, Rosa Vasconcelos, and Filomena Soares. A Review in the Use of Artificial Intelligence in Textile Industry. In *Innovations in Mechatronics Engineering*, pages 377–392. Springer, 2022.

[11] Christos Anagnostopoulos, Dimitrios Vergados, Eleftherios Kayafas, Vassilis Loumos, and George Stassinopoulos. A computer vision approach for textile quality control. *The Journal of Visualization and Computer Animation*, 12(1):31–44, 2001.

[12] Anirbid Sircar, Kriti Yadav, Kamakshi Rayavarapu, and Namrata Bist. Application of Machine Learning and Artificial Intelligence in Oil and Gas Industry. *Petroleum Research*, 6(4):379–391, 2021.

[13] Janmenjoy Nayak, Kanithi Vakula, Paidi Dinesh, Bighnaraj Naik, and Danilo Pelusi. Intelligent Food Processing: Journey From Artificial Neural Network to Deep Learning. *Computer Science Review*, 38:100297, 2020.

[14] Vijay Kakani, Van Huan Nguyen, Basivi Praveen Kumar, and Hakil Kim. A Critical Review on Computer Vision and Artificial Intelligence in Food Industry. *Journal of Agriculture and Food Research*, 2:100033, 2020.

[15] Rajnish Kler, Ghada Elkady, Kantilal Rane, and Abha Singh. Machine Learning and Artificial Intelligence in the Food Industry: A Sustainable Approach. *Journal of Food Quality*, 2022:1–9, 2022.

[16] Estela Ruiz, Diego Ferreño, Miguel Cuartas, Lara Lloret, Pablo M Ruiz del Árbol, et al. Machine Learning Methods for the Prediction of the Inclusion Content of Clean Steel Fabricated by Electric Arc Furnace and Rolling. *Metals*, 11(6):914, 2021.

[17] Michael E Scaman and Laertis Economikos. Computer Vision for Automatic Inspection of Complex Metal Patterns on Multichip Modules (Mcm-D). *IEEE Transactions on Components, Packaging, and Manufacturing Technology: Part B*, 18(4):675–684, 1995.

[18] Fabrice Chilly-Ngamaleu Piengang, MBA PMP, et al. Introduction to Predictive Maintenance Based Machine Learning With Risk and Uncertaincies: A State of the Art. In *Proceedings of the International Annual Conference of the American Society for Engineering Management.*, pages 1–11. American Society for Engineering Management (ASEM), 2022.

[19] Travis E Oliphant et al. *A Guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[20] Wes McKinney et al. Pandas: A Foundational Python Library for Data Analysis and Statistics. *Python for high performance and scientific computing*, 14(9):1–9, 2011.

[21] John D Hunter. Matplotlib: A 2D Graphics Environment. *Computing in science & engineering*, 9(03):90–95, 2007.

[22] François Chollet et al. Keras. https://keras.io, 2015.

[23] Charu C Aggarwal et al. Neural Networks and Deep Learning. *Springer*, 10(978):3, 2018.

[24] Jose Manuel Gutiérrez et al. *Redes Neuronales y Probabilísticas en las Ciencias Atmosféricas*. Monografías del Instituto Nacional de Meteorología. Ministerio de Medio Ambiente, Madrid, 2004.

[25] Sagar Sharma, Simone Sharma, and Anidhya Athaiya. Activation Functions in Neural Networks. *Towards Data Sci*, 6(12):310–316, 2017.

[26] Sebastian Ruder. An Overview of Gradient Descent Optimization Algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[27] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980, 2015.

[28] Larry Medsker and Lakhmi C Jain. *Recurrent Neural Networks: Design and Applications*. CRC press, 1999.

[29] Mike Schuster and Kuldip K Paliwal. Bidirectional Recurrent Neural Networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[30] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural computation*, 9(8):1735–1780, 1997.

[31] Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Samuel Harford. Multivariate LSTM-FCNs for Time Series Classification. *Neural networks*, 116:237–245, 2019.

[32] Haşim Sak, Andrew Senior, and Françoise Beaufays. Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition. *arXiv preprint arXiv:1402.1128*, 2014.

[33] Alex Graves. Generating Sequences With Recurrent Neural Networks. *arXiv preprint arXiv:1308.0850*, 2013.

[34] Christopher Olah. Understanding LSTM Networks. http://colah.github.io/posts/2015-08-Understanding-LSTMs/, 2015.

[35] Keiron O'Shea and Ryan Nash. An Introduction to Convolutional Neural Networks. *arXiv preprint arXiv:1511.08458*, 2015.

[36] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[37] Jorge Baño Medina et al. Deep Convolutional Neural Networks for Statistical Downscaling of Climate Change Projections. 2021.

[38] Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, and Turker Ince. 1D Convolutional Neural Networks and Applications: A Survey. *Mechanical systems and signal processing*, 151:107398, 2021.

[39] TensorFlow Agents. Introduction to RL and Deep Q Networks. https://github.com/tensorflow/agents/blob/master/docs/tutorials/0_intro_rl.ipynb, 2023.

[40] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement Learning: A Survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[41] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy Gradient Methods for Reinforcement Learning With Function Approximation. *Advances in neural information processing systems*, 12, 1999.

[42] Ivo Grondman, Lucian Busoniu, Gabriel AD Lopes, and Robert Babuska. A Survey of Actor-Critic Reinforcement Learning: Standard and Natural Policy Gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1291–1307, 2012.

[43] Alessandro Lazaric, Marcello Restelli, and Andrea Bonarini. Reinforcement Learning in Continuous Action Spaces Through Sequential Monte Carlo Methods. *Advances in neural information processing systems*, 20, 2007.

[44] Matthew E Taylor, Shimon Whiteson, and Peter Stone. Comparing Evolutionary and Temporal Difference Methods in a Reinforcement Learning Domain. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1321–1328, 2006.

[45] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, and Alex Graves. Playing Atari With Deep Reinforcement Learning. *arXiv preprint arXiv:1312.5602*, 2013.

[46] Christopher JCH Watkins and Peter Dayan. Q-Learning. *Machine learning*, 8:279–292, 1992.

[47] Beakcheol Jang, Myeonghwi Kim, Gaspard Harerimana, and Jong Wook Kim. Q-Learning Algorithms: A Comprehensive Classification and Applications. *IEEE Access*, 7:133653–133667, 2019.

[48] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, et al. Deep Q-Learning From Demonstrations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[49] Jianqing Fan, Zhaoran Wang, Yuchen Xie, and Zhuoran Yang. A Theoretical Analysis of Deep Q-Learning. In *Learning for Dynamics and Control*, pages 486–489. PMLR, 2020.

[50] Melanie Coggan. Exploration and Exploitation in Reinforcement Learning. *Research supervised by Prof. Doina Precup, CRA-W DMP Project at McGill University*, 2004.

[51] John Tinsley Oden and Junuthula Narasimha Reddy. *An Introduction to the Mathematical Theory of Finite Elements*. Courier Corporation, 2012.

[52] Barna Szabó and Ivo Babuška. Finite Element Analysis: Method, Verification and Validation. 2021.

[53] Jian-Ming Jin. *The Finite Element Method in Electromagnetics*. John Wiley & Sons, 2015.

[54] ANSYS Mechanical. https://www.ansys.com/products/structures/ansys-mechanical. Accessed: June 12, 2023.

[55] Alexander Kaszynski. pyansys: Python Interface to MAPDL and Associated Binary and ASCII Files, August 2020.

[56] PyAnsys Documentation. https://docs.pyansys.com/version/stable/. Accessed: June 12, 2023.

[57] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[58] Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Samuel Harford. Multivariate LSTM-FCNs for Time Series Classification, 2018.

[59] Jie Hu, Li Shen, and Gang Sun. Squeeze-And-Excitation Networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[60] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, et al. The UCR Time Series Classification Archive, July 2015. www.cs.ucr.edu/~eamonn/time_series_data/.

[61] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification, 2015.