Universitat Autònoma
de Barcelona

A Thesis for the

Master in Telecommunication Engineering

LSTM Networks for the Implementation of an Internal Model Control Strategy in WWTPs

by

Ricardo Colin Pérez

Supervisors: Ivan Pisa Dacosta and José López Vicario

Department of Telecommunications and Systems Engineering

Escola d'Enginyeria
Universitat Autònoma de Barcelona (UAB)

July 2020

UAB

Els sotasignants, Ivan Pisa Dacosta i José López Vicario, Professors de l'Escola d'Enginyeria de la Universitat Autònoma de Barcelona (UAB),

CERTIFIQUEN:

Que el projecte presentat en aquesta memòria de Treball Final de Màster ha estat realitzat sota la seva direcció per l'alumne Ricardo Colin Pérez.

I, perquè consti a tots els efectes, signen el present certificat.

Bellaterra, 10 de juliol de 2020.

Signatures: Ivan Pisa Dacosta José López Vicario

Resum

Els recursos hídrics són un dels assumptes ambientals més importants degut a la vulnerabilitat de l'aigua a ser contaminada i el seu efecte directe en la salut humana. La millora de la seva gestió depèn en gran mesura del tractament d'aigües residuals. Per aquest motiu, hi ha estrictes límits de contaminació establerts per l'efluent de les estacions depuradores d'aigües residuals (EDAR), el qual és abocat en rierols o altres aigües receptores. Amb la finalitat de millorar la qualitat del tractament, complir les normes imposades per les autoritats i mantenir un baix cost de les operacions, s'apliquen estratègies de control en aquestes estacions.

Aquest projecte presenta un sistema de control basat en controladors per model intern (IMC) que empren xarxes neuronals artificials (ANN), com alternativa a l'estratègia de control per defecte del Benchmark Simulation Model no. 1 (BSM1), un marc que emula el comportament d'una EDAR de propòsit general que utilitza controladors proporcionals integrals (PI). Amb el mètode de control proposat, el comportament real de l'estació es modela amb només les dades de l'influent i de l'efluent per tenir sota control les concentracions d'oxigen dissolt i de nitrat i nitrit de nitrogen, oferint també un millor rendiment en termes de la integral de l'error absolut (IAE) i de la integral de l'error quadràtic (ISE) respecte als controladors per defecte.

Resumen

Los recursos hídricos son uno de los asuntos ambientales más importantes debido a la vulnerabilidad del agua a ser contaminada y su efecto directo en la salud humana. La mejora de su gestión depende en gran medida del tratamiento de aguas residuales. Por este motivo, hay estrictos límites de contaminación establecidos para el efluente de las estaciones depuradoras de aguas residuales (EDAR), el cual es vertido en arroyos u otras aguas receptoras. Con el fin de mejorar la calidad del tratamiento, cumplir las normas impuestas por las autoridades y mantener un bajo costo de las operaciones, se aplican estrategias de control en estas estaciones.

Este proyecto presenta un sistema de control basado en controladores por modelo interno (IMC) que emplean redes neuronales artificiales (ANN), como alternativa a la estrategia de control por defecto del Benchmark Simulation Model no. 1 (BSM1), un marco que emula el comportamiento de una EDAR de propósito general que utiliza controladores proporcionales integrales (PI). Con el método de control propuesto, el comportamiento real de la estación se modela con sólo los datos del influente y del efluente para tener bajo control las concentraciones de oxígeno disuelto y de nitrato y nitrito de nitrógeno, ofreciendo también un mejor rendimiento en términos de la integral del error absoluto (IAE) y de la integral del error cuadrático (ISE) con respecto a los controladores por defecto.

Abstract

Water resources are one of the most important environmental issues due to the vulnerability of water to contamination and its direct effect on human health. Improving their management depends largely on the treatment of wastewater. For that reason, there are strict pollution limits set for the effluent of wastewater treatment plants (WWTPs), which is discharged into streams or other receiving waters. In order to improve the treatment quality, meet the standards imposed by authorities and to maintain low cost of operations, control strategies are implemented in these plants.

This project presents a control system based on internal model controllers (IMCs) adopting artificial neural networks (ANNs), as an alternative to the default control strategy of the Benchmark Simulation Model no. 1 (BSM1), a framework emulating the behavior of a general purpose WWTP that uses proportional integral (PI) controllers. With the proposed control approach, the real plant behavior is modeled with only influent and effluent data to take under control the dissolved oxygen and nitrate and nitrite nitrogen concentrations, also offering a better performance in terms of integral absolute error (IAE) and integral square error (ISE) with respect to the default controllers.

Agradecimientos

Una vez llegados al final de esta disertación, es momento de reconocer el apoyo de quienes colaboraron de alguna forma u otra en su desarrollo. Porque sin ellos, este trabajo no sería el mismo.

En primer lugar, quiero agradecer la dedicación y el compromiso de mis directores, quienes han sido indispensables para la realización de este proyecto. Gracias Ivan Pisa Dacosta por confiar en mí, por la motivación y por poder contar con tu ayuda siempre que la he necesitado; es un orgullo ser tu primer proyectista. Gracias José López Vicario por darme la oportunidad de realizar este trabajo, por tu apoyo y por tus consejos. Gracias a vosotros he descubierto el apasionante mundo del *deep learning*.

Mi agradecimiento también para mis hermanos, en especial para mi gemelo Xavi, con quien tengo la suerte de compartir tantas aventuras, incluida la de este máster. La afinidad que tenemos lo convierte en el mejor compañero de prácticas. Cómo no, agradecer el soporte de Marta, su paciencia y su comprensión durante esta etapa tan intensa. Por último, pero no menos importante, también quiero agradecer a mis padres por hacerme más fácil el camino, no solo durante la elaboración de este trabajo sino en todos los aspectos de la vida.

Ricardo Colin Pérez
24 de junio de 2020

Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Network
ANN_{dir}	Artificial Neural Network of the Controlled Process
ANN_{inv}	Artificial Neural Network of the Inverse of the Controlled Process
API	Application Programming Interface
ASM1	Activated Sludge Model no. 1
BSM1	Benchmark Simulation Model no. 1
BSM1LT	Benchmark Simulation Model no. 1 Long-term
BSM2	Benchmark Simulation Model no. 2
COST	European Cooperation in Science and Technology
DO	Dissolved Oxygen
FFNN	Feedforward Neural Network
FFT	Fast Fourier Transform
FL	Fuzzy Logic
IAE	Integral Absolute Error
IMC	Internal Model Control
ISE	Integral Square Error
IWA	International Water Association
$K_L a_x$	Oxygen Transfer Coefficient of the x Reactor Tank/BSM1 Unit
LSTM	Long Short-term Memory
MAAPE	Mean Arctangent Absolute Percentage Error

MAPE	Mean Absolute Percentage Error
MI	Mutual Information
MLP	Multilayer Perceptron
MPC	Model Predictive Control
MSE	Mean Squared Error
NARX	Nonlinear Autoregressive Exogenous
NRMSE	Normalized Root Mean Squared Error
PCA	Principal Component Analysis
PI	Proportional Integral
PPMCC	Pearson Product-moment Correlation Coefficient
Q_e	Effluent Flow Rate
Q_{in}	Influent Flow Rate
Q_{intr}	Internal Recirculation Flow Rate
Q_r	External Recirculation Flow Rate
R^2	Coefficient of Determination
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
$S_{ALK,x}$	Alkalinity of the x Reactor Tank/BSM1 Unit
$S_{I,x}$	Soluble Inert Organic Matter of the x Reactor Tank/BSM1 Unit
$S_{ND,x}$	Soluble Biodegradable Organic Nitrogen of the x Reactor Tank/BSM1 Unit
$S_{NH,x}$	Ammonium and Ammonia Nitrogen of the x Reactor Tank/BSM1 Unit
$S_{NO,x}$	Nitrate and Nitrite Nitrogen of the x Reactor Tank/BSM1 Unit
$S_{O,x}$	Dissolved Oxygen of the x Reactor Tank/BSM1 Unit
$S_{S,x}$	Readily Biodegradable Substrate of the x Reactor Tank/BSM1 Unit
TSS_x	Total Suspended Solids of the x Reactor Tank/BSM1 Unit
WWTP	Wastewater Treatment Plant
$X_{B,H,x}$	Active Heterotrophic Biomass of the x Reactor Tank/BSM1 Unit
$X_{ND,x}$	Particulate Biodegradable Organic Nitrogen of the x Reactor Tank/BSM1 Unit
$X_{S,x}$	Slowly Biodegradable Substrate of the x Reactor Tank/BSM1 Unit

Chapter 1

Introduction

Nowadays, the population growth along with the increase of industrial activity or the expansion of residential areas among others, has turned the environmental contamination into a serious threat. On the one hand, one of the most notable phenomena directly related to environmental pollution is the climate change and all its derivatives, such as rising temperatures, extreme weather events, shifting wildlife populations and habitats, rising seas, and a range of other impacts [Cli]. On the other hand, pollution not only has environmental consequences, but also effects on human health potentially leading to illnesses. That is why all actions and measures that contribute to stopping the problem of environmental contamination are welcomed.

Among the possible means of contamination such as air, soil, and water, the last-mentioned can result in human health problems. This is motivated due to its capacity to transmit dangerous diseases such as cholera and typhoid. Other effects are related to ecosystem damages produced by an excess of nutrients. They fuel algae blooms creating low-oxygen areas, which in turn can affect the aquatic life [Nun20].

There are many things people can do to contribute to the global matter of contamination, which concerns everybody. An important one consists in the reduction of the pollutant concentrations present in wastewater, and thus preserve the natural resources and environments where treated water is discharged.

Wastewater treatment plants (WWTPs) are industries devoted to treating wastewater and remove contaminants by means of highly complex and non-linear processes. They convert the wastewater into an effluent that can be returned to the water cycle (discharged into streams or other receiving waters) with minimum impact on the environment, or directly reused. This set of mechanical (physical), biological, and chemical operations done to clean wastewater allows to see WWTPs as large non-linear systems.

Moreover, since water's pollutant products are harmful to the environment at high concentrations, the treated water, which has passed through pollutant reduction processes (also generating high-pollutant products), must meet the standards imposed by authorities to ensure a certain quality. For instance, the European Directive 91/271 "Urban wastewater" [Com91] established by the European Union is a legal requirement that penalize the violation of the pollution effluent limits by means of high economic punishments to the WWTP.

For what supposes to overcome the established limits, both for fines to WWTPs and for environmental problems, control strategies have been developed to maintain the pollutants under said limits.

Figure 1.1: Satellite view of the WWTP of Montcada i Reixac.

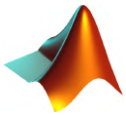
On the other side, artificial intelligence (AI) has evolved until such point that this area of computer science, solves many different types of everyday problems that are hard to tackle by human beings but relatively straightforward for computers [Goo16].

Artificial neural networks (ANNs) are one of the many methods embraced by AI that can model complex and non-linear systems, such as those performed in a WWTP, considering only the plant's influent and effluent data. This feature of ANNs is the key point of this project together with the structure of internal model controllers (IMCs), which are built upon a representation of the process under control and its inverse. The potential of ANNs to generate accurate mathematical models of high-complexity processes gives the possibility to present a new control approach based on IMC controllers adopting ANNs with excellent performance [Pis19a].

In the present dissertation two ANN-IMC controllers will be designed using long short-term memory (LSTM) networks (a type of ANN) to track two important WWTP concentrations: the first one devoted to controlling the dissolved oxygen, and the second one devoted to controlling the nitrate and nitrite nitrogen. Both controllers will be implemented in a simulation scenario replicating the WWTP's real behavior: the Benchmark Simulation Model no. 1 (BSM1). Their performance will be determined by the integral absolute error (IAE), and the integral square error (ISE) metrics, which will also be used to compare the proposed controllers with those implemented by default in the BSM1 of proportional integral (PI) type. Finally, the separately created controllers will be merged to have a WWTP scenario exclusively managed with ANNs.

The necessary tools for the development of this work are listed below. Its use throughout the project will be specified in detail in the organization of the thesis.

- **MATLAB**: it is a general, high-performance language for technical computing [MAT]. Besides, the add-on software product Simulink will be used to work with the BSM1.



- **Python**: it is a widely used high-level, general-purpose, interpreted, dynamic programming language [Pyt]. Specifically, the TensorFlow library will be used together with Keras, which is a high-level neural networks application programming interface (API) [Ker]. This last library is more user-friendly and easier to use as compared to TensorFlow [Nai17].

1.1 Motivation and Objectives

WWTP are industries focused on minimizing pollutant concentrations of residual urban waters and can therefore be considered as a mitigation measure of the environmental pollution, one of the most serious global challenges because of all the consequences it entails, some of the more remarkable presented in the introduction. For this reason, these plants must operate continuously, fulfilling the increasingly strict regulations.

Control strategies are employed to support the pollutants removal tasks, which perform highly complex biological and biochemical processes, in order to improve the treated water quality, meet the standards imposed by authorities and to decrease the costs of operation [Con13]. There are many control strategies that have been proposed with the aim of reducing the pollutant levels, so it is interesting to further suggest new control approaches that improve current solutions, which in turn lower economical costs of WWTPs and environmental contamination.

In addition, the present dissertation proposes the deployment of controllers built with ANNs, a feature that makes these systems more attractive for several reasons. The first and most obvious is that, as stated in the introduction, they can offer better performance than at least

the controllers installed by default on the BSM1, resulting in enhanced effluent quality at lower costs.

Furthermore, ANNs can model the direct and inverse processes of non-linear systems such as WWTPs, also dealing with the shortcoming of the standard IMC for processes that are not fully invertible for presenting non-linearities that are non-invertible. Besides, it is a drawback to acquire the mathematical models for the direct and inverse processes.

Another valuable benefit of working with ANNs is that they are data-driven methods, which means that they can define a model of a WWTP with only the plant's influent and effluent data, making ANN-IMC controllers a low-cost alternative to expensive hardware, and without needing any knowledge about the process behavior. This last fact motivates the adoption of ANNs in this kind of industries. At the same time, it becomes more interesting along with the term of big data [O'L13]. This ingredient, together with the current computing power, are the core of deep learning and ANNs because their models improve with the amount of data. A good analogy is that the rocket engine is the deep learning models and the fuel is the huge amounts of data we can feed to these algorithms [Ng15].

In summary, the main contribution of this work, driven by the basic motivation explained above, can be broken down into the following objectives:

- Design of an IMC controller based on ANNs to enhance the default control of the dissolved oxygen of the fifth reactor tank ($S_{O,5}$): achieve better IAE and ISE values than the BSM1 PI controller of the $S_{O,5}$.
- Design of an IMC controller based on ANNs to enhance the default control of the nitrate and nitrite nitrogen of the second reactor tank ($S_{NO,2}$): achieve better IAE and ISE values than the BSM1 PI controller of the $S_{NO,2}$.
- Establishment of a control system exclusively using IMC controllers based on ANNs to enhance the default control strategy: achieve better IAE and ISE values than the BSM1 PI controllers.

The next section describes the organization of this dissertation, and as will be seen, each of the above lines is dedicated to an individual section of Chapter 4.

1.2 Thesis Outline

After an initial introduction to control systems based on ANNs in the context of WWTPs, as well as the motivation and objective surrounding this project, the present section provides an outline of the dissertation. The topics covered are depicted in the diagram of Figure 1.2.

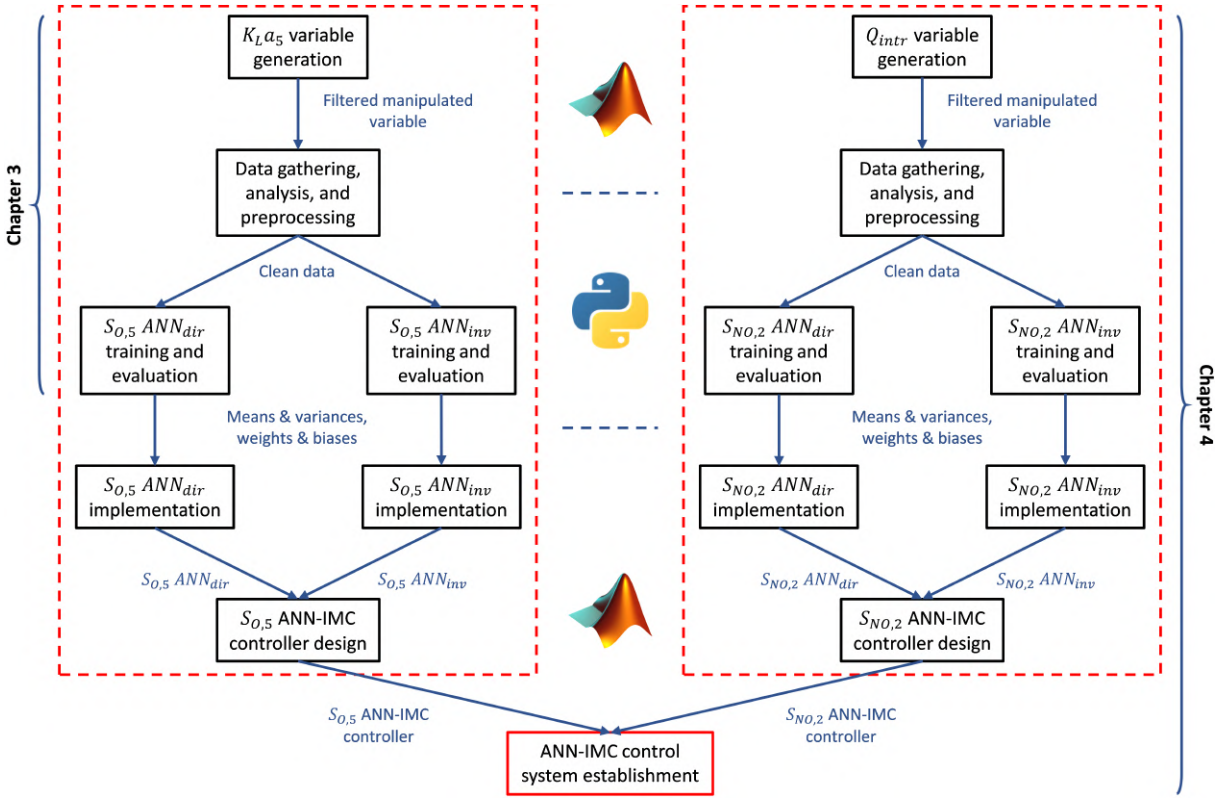


Figure 1.2: Diagram of the thesis outline.

Chapter 2

The next chapter will provide a brief description of the state-of-the-art, together with the theoretical concepts necessary for the elaboration and comprehension of this work. For this purpose, first the most innovative strategies for WWTP control will be discussed. After that, the concepts related to the working scenario and the control strategy of this project will be outlined. In summary, this last part will comprise an introduction to BSM1 and the fundamentals of the two key elements of the proposed control strategy: IMC controllers and ANNs (LSTM structures).

Chapter 3

In this chapter, a practical view of the cycle for designing neural networks will be provided. This workflow will be oriented to the ANNs that will form the IMCs of the next chapter. As shown in Figure 1.2, a total of four networks will be created to implement the two controllers. For the sake of brevity, this episode attempts to deepen in the part of the design of these ANNs, so that it can be extrapolated in Chapter 4, which will concentrate mainly on their configuration and implementation in the BSM1.

Following the schematic of Figure 1.2, it will be seen how the manipulated variables are generated and filtered to obtain raw data. Then, the preprocessing applied to the collected and

analyzed data will be presented. This first part, as can be observed in the diagram of the thesis organization, will be carried out mainly in MATLAB and Simulink, although the analysis and preprocessing of the data will be done with Python for convenience. When the data is ready to be used, it will be shown how the networks that form the controller are built, again using Python.

Chapter 4

This chapter focuses on the design of the two ANN-IMC controllers and their establishment in the BSM1 framework. The first one is designed in Section 4.1 and is devoted to control the $S_{O,5}$. Its construction is made up of multiple stages, the main ones shown in Figure 1.2. Once the ANNs are built following the steps of Chapter 3, the parameters that define the ANN architectures (weights and biases), along with the means and variances, will be exported so that the networks can be implemented in the environment in which the BSM1 is provided: MATLAB and Simulink. Finally, all that will remain is to put together these pieces created separately, which form this innovative IMC concept, in order to improve the default $S_{O,5}$ monitoring of the BSM1.

The second controller, in charge of track the $S_{NO,2}$, is developed in Section 4.2. Here, the problem addressed is similar to the one of Section 4.1 in terms of procedure. However, this challenge is completely different in terms of design, since as it will be seen, the process to be controlled in this case is totally different from the one of the $S_{O,5}$. This implies starting from scratch in the searching for the best configuration of the networks and the most appropriate data to model the direct and inverse processes that conform an $S_{NO,2}$ ANN-IMC controller. Therefore, greater results than those initially reported by the simulation benchmark will be obtained.

Before specifying the overall conclusions and possible topics for future research in Chapter 5, Section 4.3 takes the two controllers created independently in the preceding sections to join them in the same simulation framework, and achieve an enhanced control, uniquely from data, of a scenario that replicates the behavior of a general purpose WWTP.

Chapter 2

Overview of the Working Scenario and the ANN-IMC Control Approach

The aim of this chapter is to provide an overview of the most advanced developments in controllers for industrial applications, and to cover the necessary theoretical aspects to fulfill with the goals of this dissertation. To this end, Section 2.1 presents the most recent stage in the history of controllers applied in factories such as wastewater treatment plants (WWTPs), also incorporating the newest ideas and features. Section 2.2 introduces the working scenario with an explanation of the main components of its layout that emulate a WWTP, emphasizing on its control strategy and the criteria that will be used to determine the yield of the diverse control strategies. The structure of the internal model controllers (IMCs) considered in this dissertation as well as its fundamentals are presented in Section 2.3. Finally, in Section 2.4, basics of deep learning are exposed from a general view such as artificial intelligence (AI), to a specific view such as long short-term memory (LSTM) structures (the type of artificial neural networks (ANNs) that will be used in the subsequent chapters).

2.1 Control Trends in the WWTP Industry

Over the last few years, the field of artificial intelligence has progressed spectacularly due to the repercussions of the increasing big data in conjunction with the current computing power. This makes the state-of-the-art of this technology difficult to define, as in a short period of time it has shown a great development. Specifically, this study is focused on deep learning and artificial neural networks, which is one of the subsets of AI that is generating more interest because of

its capacity to get closer to human perceptive power.

From the control point of view, the Benchmark Simulation Models no. 1 and 2 (BSM1 and BSM2) representing general purpose WWTP architectures, implement default control strategies based on proportional integral (PI) controllers to maintain the dissolved oxygen (S_O), and the nitrate and nitrite nitrogen (S_{NO}) concentrations of certain reactor tanks by modifying the oxygen transfer coefficient (K_La), and the internal recirculation flow rate (Q_{intr}), respectively [Ale08, Jep07]. The BSM1 scenario is the starting point of this thesis since, as mentioned in the introduction, all its objectives turn around implementing a control strategy based on neural networks that enhances the operation of the BSM1 controllers.

Another use of this technology in the context of this work that should be highlighted is the adoption of the ANN-based internal model control strategy in [Pis19a]. Here, the proposed control strategy is deployed over the BSM1 framework to control the concentration of the dissolved oxygen in the fifth reactor tank ($S_{O,5}$) with multilayer perceptron (MLP) neural networks. This paper can be considered as the precursor of the present dissertation. However, the results of this research will be improved by adopting another kind of ANNs that are highly advisable for time series problems: long short-term memory (LSTM) networks. Moreover, this work will go further by proposing an ANN-IMC version of the other BSM1 controller (devoted to tracking the concentration of the nitrate and nitrite nitrogen of the second reactor tank ($S_{NO,2}$)), and bringing the two new structures together in a single simulation scenario.

In addition, other control approaches improving the default strategy of BSM1 and BSM2 such as model predictive (MPC) or fuzzy logic (FL) controllers are shown in [She08, Bar18]. Nevertheless, the design of these methods is more complex than the straightforward IMC schematic implemented in this dissertation, which is just based on available measured data. At the same time, the proposed approach exploits the benefit of ANNs when modeling complex non-linear functions.

Finally, there are further applications outside the WWTP industry where ANNs have been used in the IMC implementation to control different processes such as the light of an office or the movement of a vehicle [Kan18, Wan18]. Nevertheless, these works also adopt feedforward neural networks (FFNNs) instead of recurrent neural networks (RNNs) like LSTMs to design the ANN-IMC control approach.

In short, the strategy implemented in this project replaces the main components of the traditional IMC schematic, i.e., the model of the process under control and its inverse, by LSTM neural networks. Here, the IMC concept based on this kind of networks is implemented in WWTPs, although it is not limited to be applied in other areas where predictive controllers are required and there is temporal dependence on the available data.

2.2 Benchmark Simulation Model no. 1 (BSM1)

The Benchmark Simulation Model no. 1 is a simulation scenario undertaken in Europe from 1998 to 2004 by Working Groups of COST (European Cooperation in Science and Technology) Action 682 and 624 (Alex et al., 1999) [Ale99], and now continued under the umbrella of the IWA (International Water Association) Task Group on Bechmarking of Control Strategies for WWTPs [Cop02]. This "*simulation benchmark*" replicates the WWTP's real behavior implementing the Activated Sludge Model no. 1 (ASM1) as the biological process model (Henze et al., 1987) [Hen87]. It is available for free on the Lund University's website: `https://www.iea.lth.se/WWTmodels_download/`.

The activated sludge is a process dealing with the treatment of sewage and industrial wastewater. Its aim is to achieve, at minimum costs, a sufficiently low concentration of biodegradable matter in the effluent together with minimal sludge production. To do this, the process must be controlled. In the case of BSM1, it considers a default controller strategy based on PI controllers for this purpose [Ale08]. However, many control strategies can be proposed.

Without a standardized protocol, comparison of different control strategies, either practical or based on simulation, is difficult due to several reasons, including:

- The variability of the influent.
- The complexity of the biological and biochemical phenomena.
- The large range of time constants (varying from a few minutes to several days).
- The lack of standard evaluation criteria (among other things, due to region specific effluent requirements and cost levels).

In order to deal with these differences, this simulation environment was developed as a tool for evaluating activated sludge wastewater treatment control strategies; as a "benchmark" from which to judge and compare the performance of the different control strategies.

Although in this work the BSM1 will be used as the working scenario for the sake of simplicity, two other later benchmarks versions with additional features are available:

- **BSM1**: relatively simple, it combines nitrification with predenitrification, which is most commonly used for nitrogen removal. The control strategies are evaluated over periods of 14 days, with different weather conditions [Ale08].
- **BSM1LT**: it is based on BSM1, but with a longer evaluation period (609 days). Faults (toxic events, problems on sensors and actuators) can occur.

- **BSM2**: the BSM2 layout includes BSM1 for the biological treatment of the wastewater. The sludge treatment is taken into account [Jep07].

As stated above, the BSM1 will be the working scenario. Considering that this project will seek an ANN-based alternative to the default control strategy, the simulator will play an important role. It will be used first in the data generation (influent and effluent measurements will be required to train the neural networks), and then to test the control approach based on IMC adopting ANNs once it is implemented.

Regarding some general characteristics, its influent dynamics are defined by means of files for three different weather conditions: dry weather, rain weather (a combination of dry weather and a long rain period) and storm weather (a combination of dry weather with two storm events). Each one contains 14 days of influent data with sampling intervals of 15 minutes. For initialization, a 100-day period of stabilization in closed-loop using constant inputs (average dry weather flow rate and flow-weighted average influent concentrations) with no noise on the measurements has to be completed before using the dry weather file (14 days) followed by the weather file to be tested [Ale08].

2.2.1 Plant Layout

The first plant layout (BSM1), modeling the water line of a WWTP, is composed of five reactor tanks where the biological and biochemical processes defined in ASM1 are performed. The first two reactor tanks are anoxic tanks, which means they are working in anoxic conditions (with a lack of oxygen), while the next three tanks are aerobic tanks, which means they are working under aerobic conditions (with large amount of oxygen) [Ger14]. Thus, the plant combines nitrification with predenitrification in a configuration that is commonly used for achieving biological nitrogen removal in full-scale plants [Con13].

After that, the activated sludge reactor is in series with a secondary settling tank, which is responsible for separating the activated sludge solids from the mixed liquid. This secondary clarifier is modeled as a 10-layer non-reactive unit (no biological reaction) wherein the 6^{th} layer (counting from bottom to top) is the feed layer [Ale08].

Finally, a basic control strategy with two control loops is proposed to test the benchmark:

1. **Control loop #1**: involves controlling the dissolved oxygen (DO) level in the last aerobic compartment (the fifth tank) by manipulation of the oxygen transfer coefficient.
2. **Control loop #2**: involves controlling the nitrate level in the last anoxic compartment (the second tank) by manipulation of the internal recycle flow rate.

Figure 2.1 shows a general overview of the layout.

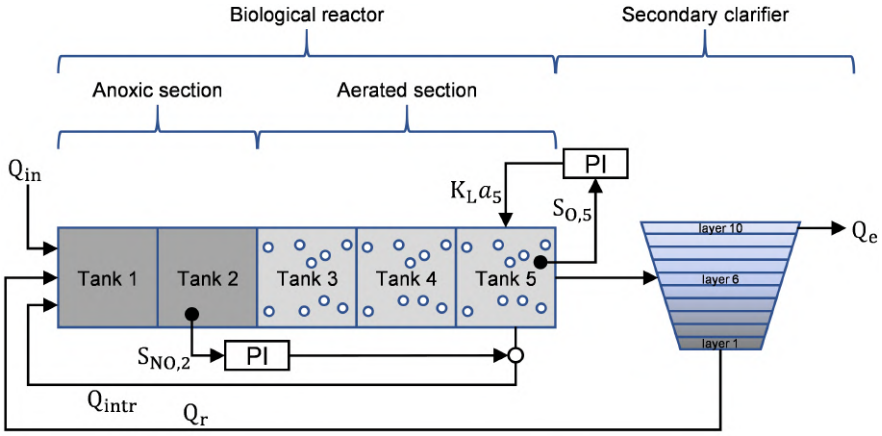


Figure 2.1: General overview of the BSM1 plant.

In terms of flow rates, this plant is designed for an average influent dry weather flow rate of 18446 m^3/day, a volume of 1000 m^3 for each anoxic tank, a volume of 1333 m^3 for each aerated tank, and a volume of 6000 m^3 for the non-reactive secondary settler. It means that the hydraulic retention time (based on average dry weather flow rate and total tank volume, i.e., biological reactor + secondary clarifier, of 12000 m^3) is 14.4 hours. Finally, the main flows are: the influent flow rate (Q_{in}), the internal recirculation flow rate (Q_{intr}) (understood as the quantity of aerated flow going from the fifth tank to the first one), the external recirculation flow rate (Q_r), and the WWTP's effluent flow rate (Q_e).

This same plant layout is represented in Simulink by the system in the figure below, where the main parts that make up the plant can be easily identified [Ale08].

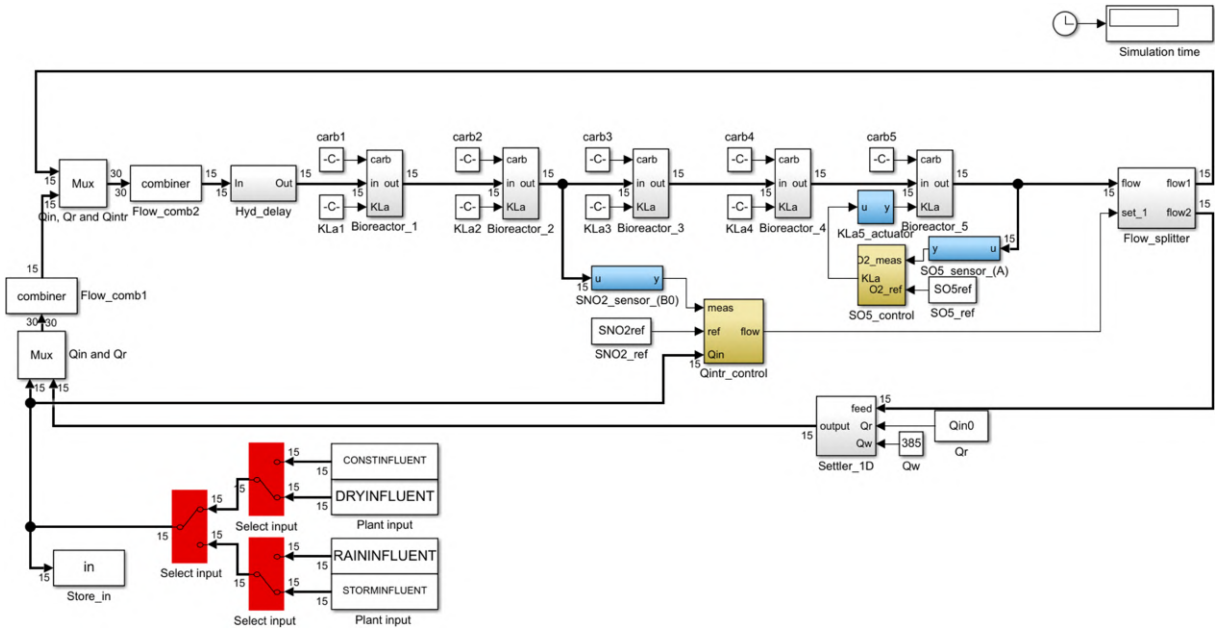


Figure 2.2: Interface layout of the BSM1 plant in Simulink.

2.2.2 Default Control Strategy

In order to enhance the water treatment process, and therefore to make it more efficient and cost-effective, control loops are implemented in WWTPs to manipulate different variables of the system. A better operation of the control loops means better effluent quality at a lower cost. By default, the control strategy of the BSM1 is based on two feedback loops defined in terms of proportional integral (PI) controllers. As stated in the previous subsection, the first one is devoted to controlling the dissolved oxygen in the fifth thank ($S_{O,5}$) by using the oxygen transfer coefficient of the fifth tank (K_La_5) as control handle, while the second one is devoted to controlling the nitrate of the second tank ($S_{NO,2}$) by using the internal recirculation flow rate (Q_{intr}) as control handle. In other words, the first controller varies the K_La_5 concentration to maintain the $S_{O,5}$ at a certain set point, while the second one varies the Q_{intr} to maintain the $S_{NO,2}$ at a certain set point.

PI controllers are by far the most used controllers in industry because of their simplicity and good trade-off between performance and robustness [Men16]. Only the two parameters that give the name to the controller need to be adjusted in the so-called controller tuning stage: the proportional gain (K_p) and the integral gain (T_i). The value of the controller output, which is fed into the system as the control handle or manipulated variable ($u(t)$), is computed as follows:

$$u(t) = K_p\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau\right) \tag{2.1}$$

where $e(t)$ represents the feedback error: $e(t) = r(t) - y(t)$, $y(t)$ being the controlled variable and $r(t)$ its desired value or set point. Then, the effort of control strategies is dedicated to achieving a controlled variable as similar as possible to the set point by means of the manipulated variable. For a better understanding, Figure 2.3 shows the PI-based control loops of the control strategy predefined within the BSM1 scenario.

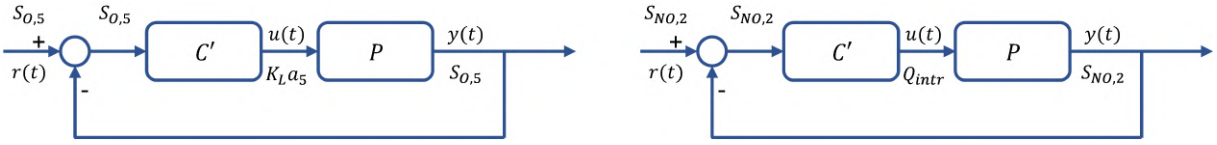


Figure 2.3: PI controller structures for $S_{O,5}$ (left) and $S_{NO,2}$ (right).

This is how the the first and second control loops keep the dissolved oxygen and nitrate concentrations in the fifth and second tanks at a predetermined level. It is achieved by manipulating the oxygen transfer coefficient and the internal recycle flow rate respectively, by means of the PI controller (C'). A sensor is used to measure the controlled variable and provide feedback to the control system. The BSM1 framework gives the possibility to measure the controlled variables by means of sensors with or without noise [Ale08]. For this work, ideal sensors will be adopted

(the scenario with noise is out of the scope of this thesis). Once measured the concentration, the difference between the controlled variable and the set point (the feedback error) is used by the controller to determine the manipulated variable to drive the real process (P). Said process represents the fifth tank for the first control loop, and the first and second tanks (among others) for the second control loop. The K_p and T_i parameters must be fixed for each one of the control loops.

Finally, as it was done for the plant layout, the design of the PI controllers in Simulink is shown in the figure below.

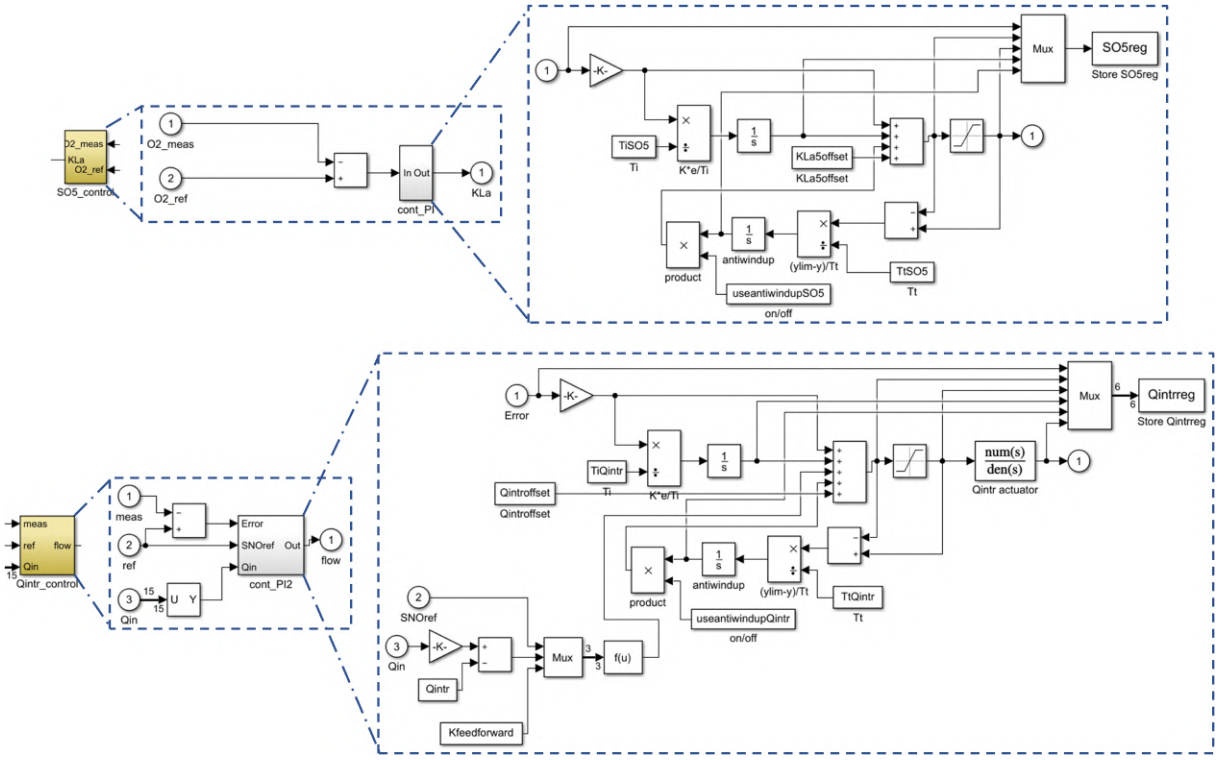


Figure 2.4: Interface layout of the PI controllers in Simulink for $S_{O,5}$ (top) and $S_{NO,2}$ (bottom).

2.2.3 System Performance Assessment

The performance assessment is made at a level concerning the control loops, evaluated by the integral of the absolute error (IAE), and the integral of the squared error (ISE) criteria. These two metrics quantify the effect of the control strategy on the controller performance. Basically, the results will serve as a proof that the proposed control strategy has been applied properly and improves the functioning of the default version. In other words, the results will help to determine how well each structure is tracking changes in the set point.

As stated earlier, the control strategies are simulated over periods of 14 days with different weather conditions. However, BSM1 simulation protocol specifies that the period of observation

for IAE and ISE is the second week or 7 last days for each weather file. Considering this statement, the expressions of the two measurements that will be used to evaluate the performance of the systems are the following:

$$IAE = \int_{t=7\ days}^{t=14days} |r(t) - y(t)|dt \tag{2.2a}$$

$$ISE = \int_{t=7\ days}^{t=14days} \big(r(t) - y(t)\big)^2 dt \tag{2.2b}$$

The difference between the controlled variable and the reference signal represents the control error, and the integrals are evaluated over the time period of the second week. Both measures require a fixed experiment to be performed on each system, i.e., a single set point to always compare metrics and strategies from the same stage.

On the one hand, the IAE integrates the absolute error over time. This measure penalizes the error linearly with the magnitude, without adding weight to any of the errors in a systems response. Consequently, the smaller the IAE the better the tracking, being zero the ideal case. On the other hand, the ISE integrates the square of the error over time. Unlike the IAE, this metric penalizes large errors more than smaller ones (since the square of a large error will be much bigger). Thus, high ISE indicates that the control system produces large errors, so values close to zero are desirable.

2.3 Fundamentals of Internal Model Control (IMC)

As seen in the previous section, the operation of a WWTP is supported by control loops that allow the manipulation of system variables like the dissolved oxygen and the nitrate of BSM1. After a brief presentation of the basics of the BSM1 default feedback loops in Subsection 2.2.2, the basics of the control strategy considered in this dissertation to improve the maintenance of process variables, i.e., the IMC, are presented in this section.

In this work, the default PI controllers are changed by IMC controllers. The philosophy of the latter relies on the internal model principle, which states that "*control can be achieved only if the control system encapsulates, either implicitly or explicitly, some representation of the process to be controlled*" [Gar82]. Particularly, if the control scheme is developed based on an exact model of the process being controlled, then perfect control is theoretically possible. It means that the performance of this kind of controllers depends directly on the process model accuracy, so the effort in Chapter 4 will be dedicated to designing the ANNs to model the processes as precisely as possible. The standard IMC control system (controller and process model) is depicted in Figure 2.5.

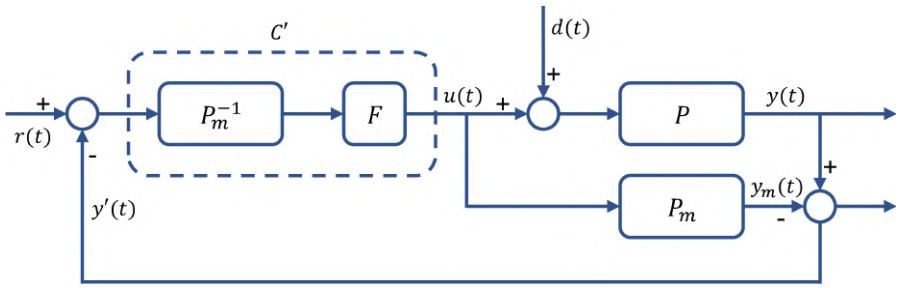


Figure 2.5: IMC controller structure.

On the one hand, P corresponds to the real process, P_m represents its model, P_m^{-1} is the inverse of P_m and F corresponds to a first-order filter. On the other hand, C' represents the IMC controller itself. Finally, using the same nomenclature as earlier for PI controllers, $r(t)$, $u(t)$, and $y(t)$ corresponds to the reference, controller, and process output, while $d(t)$, $y_m(t)$ and $y'(t)$ correspond to the perturbation, model output, and the difference between the process and model outputs (the estimated effect of disturbance), respectively.

According to Figure 2.5, the operation of IMC systems is described by the expressions below:

$$y(t) = P\big(u(t) + d(t)\big) \qquad \text{(2.3a)} \qquad\qquad u(t) = C'\big(r(t) - y'(t)\big) \qquad \text{(2.3b)}$$

where $P(\cdot)$ is the real process to control, $y'(t) = y(t) - y_m(t)$ is the effect of disturbances and of a mismatch of the model, being $y_m(t) = P_m\big(u(t)\big)$, where $P_m(\cdot)$ is the model of the real process to control. Lastly, the IMC controller behavior is described by the following equation:

$$C'(\cdot) = F\big(P_m^{-1}(\cdot)\big) \qquad (2.4)$$

where $F(\cdot)$ is a first order filter, in general a low-pass filter used to attenuate the effects of process-model mismatch [Gan13].

If the output of P_m and the output of P are the same, which means that P_m is exactly equal to P ($y'(t) = y(t) - y_m(t) = 0$), and there is no disturbance ($d(t) = 0$), the control system behaves as if it was in open loop (perfect tracking):

$$y(t) = P\big(u(t)\big) = P\Big(F\Big(P_m^{-1}\big(r(t)\big)\Big)\Big) = r(t) \qquad \text{(2.5a)}$$

$$u(t) = C'\big(r(t)\big) = F\Big(P_m^{-1}\big(r(t)\big)\Big). \qquad \text{(2.5b)}$$

Otherwise, if there is mismatch between P and P_m or if a disturbance acts on the system, the feedback loop enters into play [Roj16]. In practice, this is the common situation: process-model mismatch and unknown perturbations affecting the system.

With the IMC controller concept that this project presents, unlike traditional IMC design techniques, this approach does not require mathematical models defining P_m and its inverse P_m^{-1}. Instead, these processes are integrated into the IMC controller structure as artificial neural network (ANN) models, in such a way that they mimic the processes' behavior in an accurate manner. As will be seen in the next section, ANNs are computational models based on the structure and functions of biological neural networks. The model of the process being controlled is emulated by an ANN built to map $u(t)$ to $y_m(t)$ (ANN_{dir}), and its inverse by another ANN built to map $r(t) - y'(t)$ to the input of F (ANN_{inv}). As it will be seen in Chapter 3, these ANN models used for controlling the process are constructed uniquely with data, which is an advantage over the original PI controller. In the context of this work, perturbations (such as the different weather conditions) will be included as part of the actual process, since the ANNs will be constructed with this factor integrated into the data. Figure 2.6 presents the redesigned IMC controller structure that will be employed in this dissertation.

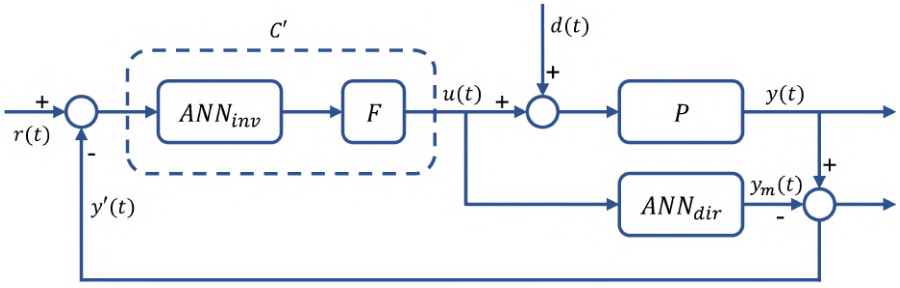


Figure 2.6: ANN-IMC controller structure.

Here, $y_m(t)$ is computed as $y_m(t) = ANN_{dir}\big(u(t)\big)$ and the IMC will be finally described as:

$$C'(\cdot) = F\big(ANN_{inv}(\cdot)\big) \tag{2.6}$$

That is why this section opens the door to the next one, where it will be understood what ANNs are, and which are the main steps for their design.

2.4 Fundamentals of Deep Learning

Before moving on to the content and results of this project, it is convenient to introduce certain basic concepts of what deep learning is, what it can achieve, and how it works. This will enable a better understanding of this thesis.

First, it is important to take into account that deep learning is a subset of machine learning, which in turn is a subset of artificial intelligence (AI) as it is shown in Figure 2.7.

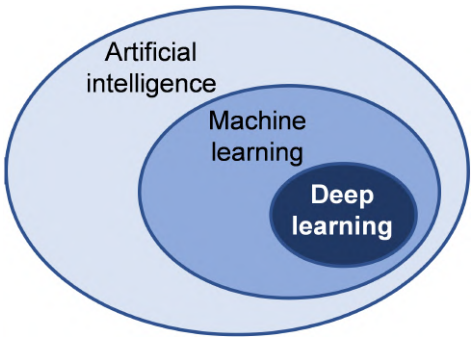


Figure 2.7: A representation of the relationship between artificial intelligence, machine learning, and deep learning.

Although deep learning is just a subset of AI, it is an important subset since their techniques apply for many applications in current life such as the widespread deployment of practical speech recognition, machine translation, self-driving cars, or image recognition among others.

Having seen how these three concepts are related, in order to understand deep learning, it is suitable to start from the outer circle to the inner one as shown in Figure 2.7, i.e., from the most general concept corresponding to AI to the interested one: deep learning.

In computer science, AI is the intelligence demonstrated by machines, in contrast to the natural intelligence displayed by humans. Colloquially, the term "artificial intelligence" is often used to describe machines (or computers) that mimic "cognitive" functions that humans associate with the human mind, such as "learning" and "problem solving" [Rus10]. For this reason, this field focuses on the effort to automate intellectual tasks normally performed by humans [Cho17].

Going deeper into this broad concept, machine learning is found as one of the several subfields of AI. It is defined as the study of computer algorithms that improve automatically through experience [Mit97]. The improvement is due to the ability of these computer algorithms to be trained (rather than explicitly programmed) and learn from data. The training consists on the process of modifying the internal parameters (or weights) of the algorithm for exposure to many examples (or input/output pairs) relevant to a task. This will allow the trained algorithm to determine the appropriate response to said task when receiving new inputs. For instance, if a daily task like the handwritten digit recognition is desired to be automated, many examples of handwritten digits by various people (as input) together with the values they represent (as output), have to be presented to the machine learning algorithm. Then, it will be able to find similar patterns to distinguish the variety of numbers [Tor18]. Similarly, and among many other examples, some organizations have the capacity to predict changes in business with properly

trained machine learning algorithms, giving the straightforward value of being able to predict the future. Since machine learning algorithms learn from data, the amount and quality of available data for building a model are important factors in how well the model learns from the task (see Chapter 3). In fact, most machine learning involves transforming the data in some sense [Zha20].

A machine learning model is the output of a machine learning algorithm run on data (or trained) [Bro20]. This corresponds to a mathematical model or a parametric function which can be modified because it has parameters for such a purpose. That is why the algorithm can adapt the function to determine the relationship between its inputs and outputs. In other words, the algorithm adjusts the model parameters in order to correct or refine this input output relationship. Learning is thus the act of setting the model parameters in order to reduce the model's error, and the learning process is training because the algorithm is trained to match the correct answer (the output) to every question offered (the input). After training, when a model is provided with an input, it will give an output based on the data with which it was trained. For the sake of clarity, Figure 2.8 shows a diagram of how these concepts are related.

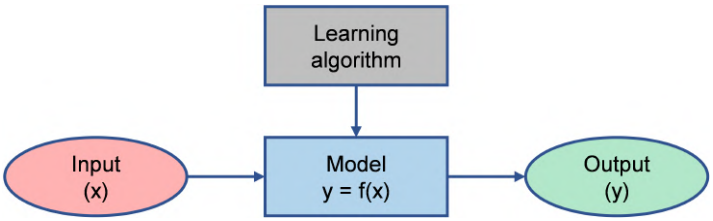


Figure 2.8: Diagram of a machine learning model.

In general, once we have the pertinent information, building a machine learning model involves the training and testing phases, as it will be seen in more detail and in a more practical view in Chapter 3.

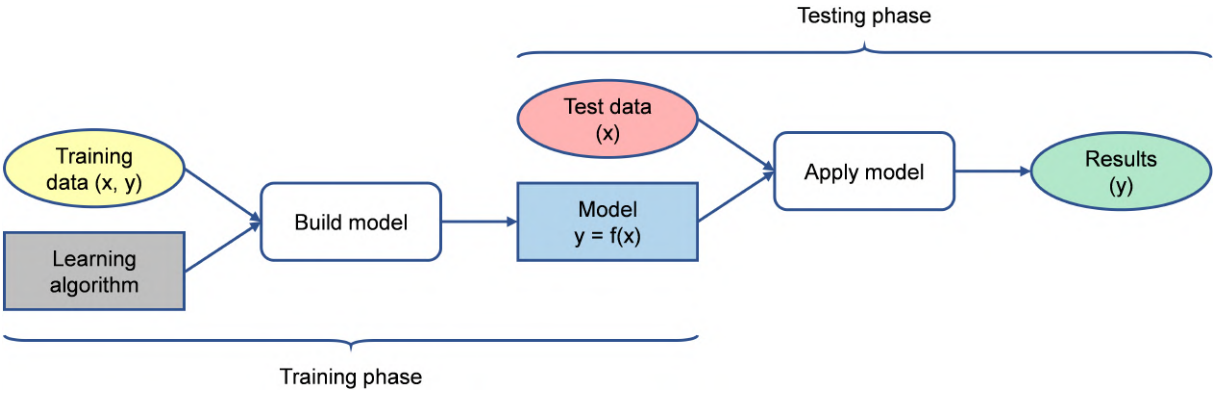


Figure 2.9: Diagram of training and testing phases.

In the training phase, the model is constructed adjusting its parameters to minimize errors by means of a learning algorithm and using a dataset for this matter, i.e., the training dataset. In the testing phase, the learned model is applied to new data (data not used in the training phase), i.e., the testing dataset, to obtain its performance. The goal in building a model is to have good operation in training, as well as in test data. Usually, an intermediate phase is carried out with the so-called validation dataset before moving on to the testing phase. It is used to determine when to stop training in order to avoid model overfitting: a problem that happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data (the opposite of underfitting, which occurs when a model can neither model the training data nor generalize to new data) [Bro19b].

Finally, the three main machine learning categories in accordance with the types of learning problems are summarized:

- **Supervised learning**: when the target (or label), which is what the model is predicting, is available (referred as having labeled data). The two main types of supervised learning problems are classification (when the target is a class), and regression (when the target is a numerical value). Some of the most popular algorithms in this category are linear regression, decision trees, or neural networks.
- **Unsupervised learning**: when the target is unknown or unavailable (referred as having unlabeled data). Two main problems that are often encountered are clustering (involves finding groups in data), and density estimation (involves summarizing the distribution of data). Some of the best-known algorithms in this category are K-means, or principal component analysis (PCA).
- **Reinforcement learning**: when the model is implemented as an agent that operates in an environment and must learn to operate using feedback [Sut18]. Some popular examples of reinforcement learning algorithms include Q-learning, temporal-difference learning, and deep reinforcement learning.

As stated in the introduction, the development of the work will be undertaken with neural networks. Besides, it will address supervised learning problems, so there will be input and output data to train these networks.

Proceeding as planned in Figure 2.7, deep learning is found a subfield of machine learning, so in both cases algorithms learn by analyzing data. However, deep learning is concerned with algorithms inspired by the structure and function of the brain called artificial neural networks (ANNs) [Bro19d].

The biggest advantage of deep learning over other machine learning subfields is the scalability [Ng15], which means that its large neural networks can deal with any amount of data, and in a nutshell, this is translated into more accuracy. Because of that, the results offered by large neural networks tend to improve as they are trained with more data, unlike other machine learning techniques which reach a plateau of performance, as represented in the figure below.

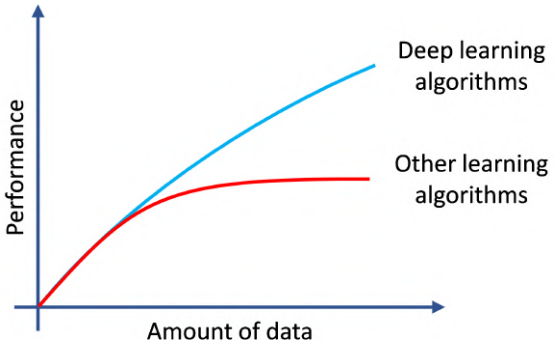


Figure 2.10: Chart representing how data science techniques scale with the amount of data.

As illustrated in Figure 2.11, an ANN consists of three or more layers: an input layer, one or more hidden layers, and an output layer. The term "deep" refers to the number of hidden layers in the ANN. Each layer has a certain number of nodes called neurons: the input layer has equal number of neurons as the dimension of input data features, the hidden layers has an arbitrary number of neurons, and the output layer has one or more neurons depending on the problem (e.g., only one if it is a regression problem, or more than one if it is a classification problem). Data is injected through the input layer, then it is modified in each of the hidden and output layers.

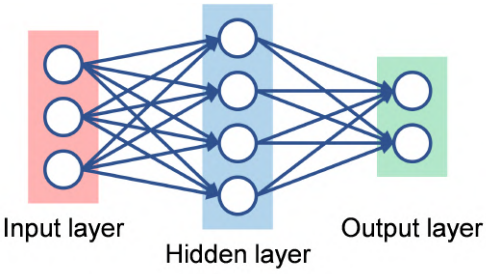


Figure 2.11: The architecture of a simple neural network.

Data modifications are based on the parameters associated to each of these nodes (the weights and biases), which determine the data modifications. In this context, learning means finding the values for the parameters of all layers in a neural network, such that the network correctly maps example inputs to their associated targets.

Basically what a single neuron (or perceptron; the simplest version of a neural network [Fre99]) does is multiply the input data by a weight vector, add a bias, and pass the result through an activation function, which is a mathematical equation, to produce a result before transferring it to further layer.

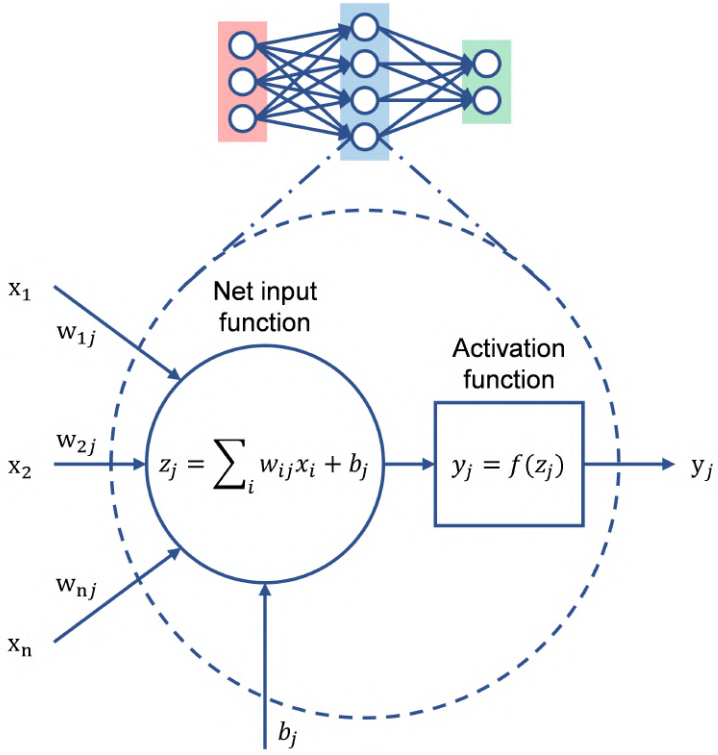


Figure 2.12: The architecture of a single neuron.

In order to find the parameters that perform these operations, an iterative process is carried out for all the known labeled examples. This is done comparing the value of their label obtained through the model with the expected value of the label of each element. After an iteration, the weights of the W and b parameters are adjusted in such a way that a loss function, in charge of measuring the quality of the network's output (the loss score), is minimized. Then, the computation of the loss function is used as a feedback signal to adjust the value of the weights through an optimizer, with the aim of reducing the loss value of the current example. This iterative procedure for finding the network parameters is summarized in the diagram in Figure 2.13.

This training loop starts with random values for the weights, resulting in outputs that are far from what they should be, producing then high loss score values. After many iterations, values are obtained for the weights that minimize the loss function, leading to outputs that are close to their labels.

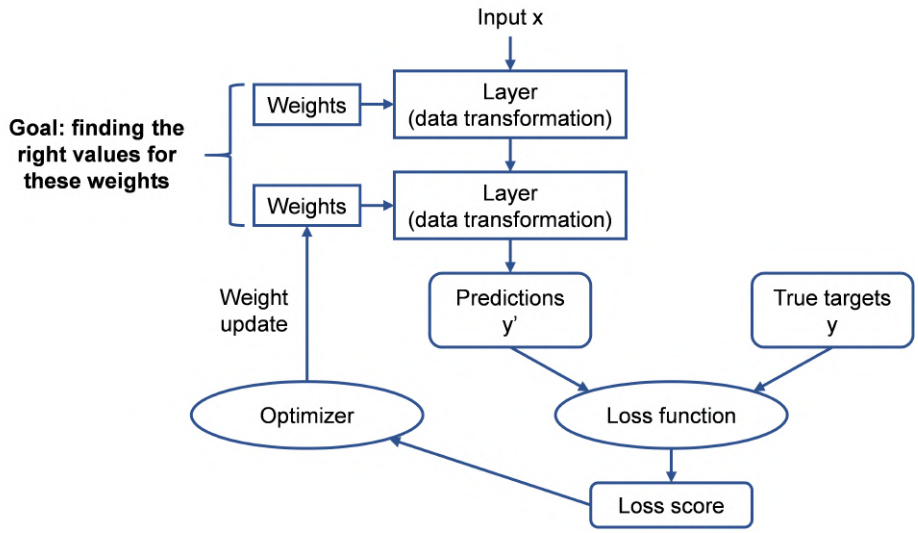


Figure 2.13: Diagram of the deep learning's iterative process.

Two last concepts that should be understood are the batch size and number of epochs, both integer values. The batch size defines the number of samples to work through before updating the internal model parameters, whilst the number of epochs defines the number of times that the learning algorithm will work through the entire training dataset. So, the size of a batch must be more than or equal to one and less than or equal to the number of samples in the training dataset, whereas the number of epochs can be set to an integer value between one and infinity [Bro19a].

There are many classes of ANNs, but to understand the ones involved in this project, i.e., recurrent neural networks (RNNs), first it is important to know the basics of feedforward neural networks (FFNNs). As it will be seen, both kind of networks are named after the way they move the information through the neurons of the network: FFNNs feed information straight through, while RNNs cycles it through a loop.

2.4.1 Feedforward Neural Networks (FFNNs)

A feedforward neural network is an ANN wherein connections between the nodes do not form a cycle [Zel94]. It means that data is moved only in forward direction, from the input neurons to the output neurons.

This class of ANNs are also called multilayer perceptrons (MLPs) and are the quintessential deep learning models [Goo16]. As its name suggests, MLPs are perceptrons (single neuron models) with more than one layer. Based on what has been seen so far, an MLP is just a mathematical function composed by many simpler functions mapping some set of input values to output values. Figure 2.11 would represent an MLP with a single hidden layer.

Because of their flexibility, MLPs are often applied to supervised learning problems: they train on a set of input/output pairs and learn to model the correlation (or dependencies) between those inputs and outputs. On the contrary, a drawback of FFNNs is that they are not designed for time-series data, what means that results with sequential data may not be so good. Unlike supervised learning problems like classification or regression, time series problems add the complexity of order or temporal dependence between observations (a time dimension). Despite the great capability offered by FFNNs, they are affected by this key of temporal dependence, almost always unknown [Bro19c].

Recurrent neural networks (RNNs), like those involved in this work, add the explicit handling of order between observations when learning a mapping function from inputs to outputs. This implies that the data processed by RNNs should be three dimensional with the shape (number of samples, number of time steps per sample, and number of variables), in contrast to FFNNs, whereby should be two dimensional with shape (number of samples, and number of variables). Instead of mapping inputs to outputs alone, RNNs can learn a mapping function for the inputs over time to an output (they also consider previous time steps), enabling them to operate better than FFNNs in these scenarios. As the data to be used in this thesis has a temporal dependence, there is an interest in using RNNs, which will be discussed in the next subsection.

2.4.2 Recurrent Neural Networks (RNNs)

An important characteristic in common of the neural networks seen so far is that they have no memory. A type of artificial neural networks that have memory are the recurrent neural networks.

Unlike FFNNs, RNNs can use their internal hidden state (memory) to process sequences of inputs. This characteristic allows RNNs to better mimic how the human brain works. When humans read a text, the words are processed one by one while keeping memories of what came before, allowing a fluid understanding of the text. It means that biological intelligence processes information incrementally while maintaining an internal model of what it is processing, built from past information and constantly updated as new information comes in [Cho17].

An RNN adopts the same idea as biological intelligence to address the memory shortcoming of FFNNs: it processes sequences by iterating through the sequence elements and maintaining a state containing information relative to what it has seen so far. On the contrary, FFNNs process the entire sequence in a single step (the sequence must be shown to the network at once) with no state between inputs.

Since RNNs take time and sequence into account, they have a temporal dimension, so they are a proper class of networks to analyze time-series and sequential data (data where the order matters) allowing to treat the dimension of “time”.

Having memory implies a type of neural network that has an internal loop so that information can persist, as is illustrated in Figure 2.14.

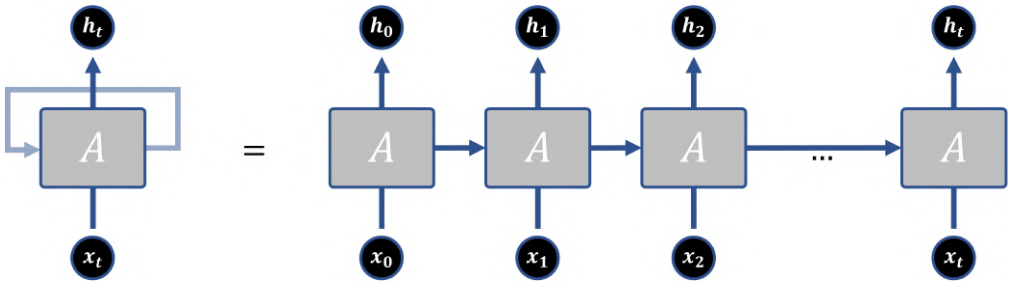


Figure 2.14: A folded and unfolded RNN cell.

The above diagram shows the folded and its equivalent unfolded version of an RNN cell, which in the most abstract setting, is anything that has a state and performs some operation that takes a matrix of inputs [RNN20]. The unfolded version makes sense of why RNNs are called "recurrent": they perform the same task for every element of a sequence, with the output being dependent on the previous computations. A chunk of neural network (A) looks at some input or sequence element (x_t) and outputs a value (h_t). A loop allows information to be passed from one step of the network to the next.

Due to the internal loop present in all RNNs in order to keep relevant information of previous inputs, they have this form of a chain of repeating modules or units of neural network (A). In standard RNNs, this repeating module has a very simple structure with a single neural network layer, as is shown in Figure 2.15.

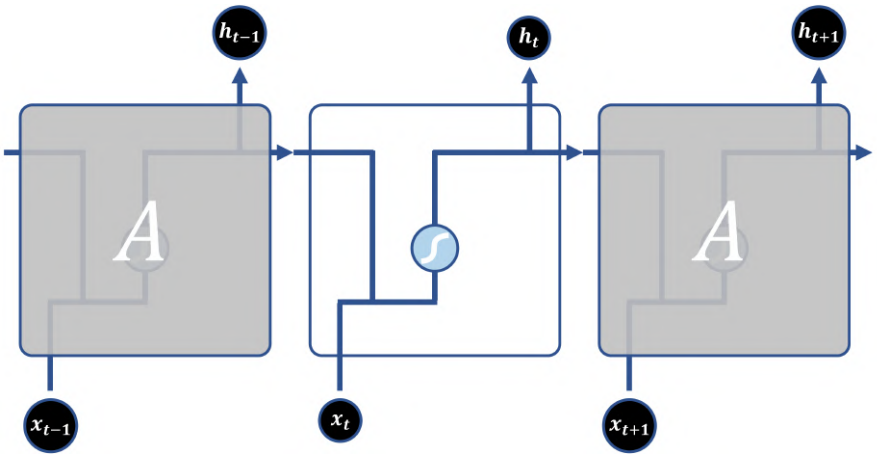


Figure 2.15: The repeating module in a standard RNN.

This sketch represents how a standard RNN loops over time steps considering at each time step the previous state (h_{t-1}) and the current input (x_t) to obtain an output or current state (h_t). The state can therefore be considered as a kind of short-term memory of a cell.

It can also be observed that the decision reached at time step t-1 (h_{t-1}) affects the decision it will reach one moment later at time step t (h_t), so the state for the next step will be the previous output unless for the first time step, where the previous output will not be defined (there will be no current state), and the state will be initialized as a zero vector.

Going a little deeper, the architecture of the repeating module in a standard RNN cell is composed by the elements forming the diagram of Figure 2.16.

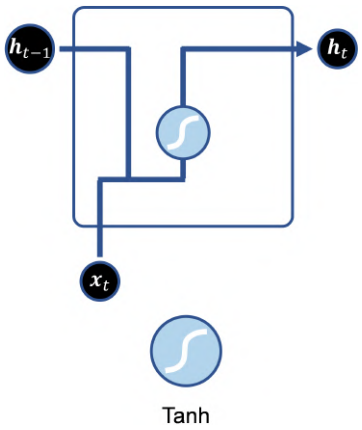


Figure 2.16: The internal structure of a standard RNN cell.

Each one of these cells comprises two sources of input: the present (x_t) and the recent past (h_{t-1}), one output (h_t), and the remaining ingredient, which is a neural network layer with hyperbolic tangent activation function. As it is explained during this section, a layer is a collection of neurons operating together at a specific depth within a neural network. There are different weights associated for the different inputs of the layer neurons. Inputs are multiplied by their respective weights, summed together with a certain bias, and finally some activation function such as the hyperbolic tangent ($tanh$) in this case is applied.

In order to get the output or current state (h_t), first, the two inputs are combined to form a vector which has information of the current (x_t) and previous inputs (h_{t-1}). After that, this vector, which is multiplied by some weights and with a certain bias added, goes through the $tanh$ activation function. It gives rise to the internal hidden state (memory) mathematical expression:

$$\boldsymbol{h}_t = tanh(\boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{b}) \tag{2.7}$$

where $\boldsymbol{x}_t$ and $\boldsymbol{h}_{t-1}$ denote the current input and the recurrent information of the previous time step, respectively. $\boldsymbol{W}$ are the weights at current input state, $\boldsymbol{U}$ are the weights at previous hidden state, and $\boldsymbol{b}$ are the biases. The $tanh$ activation function squashes values to be always between -1 and 1. It regulates the values flowing through the network, avoiding big differences between them.

However, the loop structure of a standard RNN involves performing the same previous operation for each time step. It means that internally backpropagated signals (information) passing many times through the same repeating module (through a set of stacked *tanh* layers), will tend to disappear after a few recursions considering that in the best-case scenario, the *tanh* is 1. This fact that makes a basic RNN have a short-term memory is the so-called vanishing gradient problem. It is a limitation that involves processing relatively short sequences so that previous hidden states have a relevant effect on the current output.

For this reason, standard RNNs are not a good choice for problems that require a long memory, because early signals (information) cannot be taken into account to be related with recent inputs. Most applications require an alternative designed to solve this problem, like the one used in this work: long short-term memory (LSTM) networks.

LSTM is one kind of RNN developed by Hochreiter and Schmidhuber in 1997 [Hoc97]; it was the solution to the short-term memory suffered by standard RNNs due to the vanishing gradient problem. Figure 2.17 shows that LSTMs also have a chain structure as standard RNNs, but in this case, the repeating module has a different layout. Instead of having a single neural network layer, there are four (the *tanh* on the top right represents just an activation function, not a neural network layer with a *tanh*) connected in a such a way that they allow for a larger memory, as will be seen later in detail.

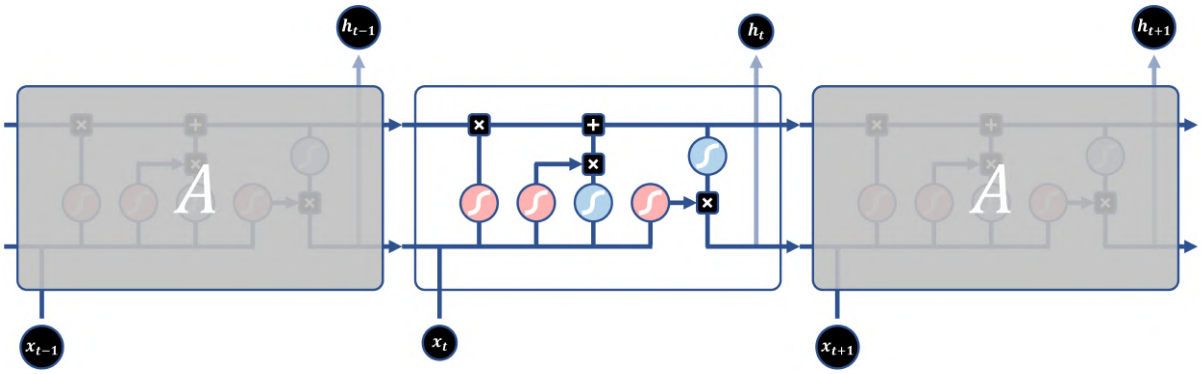


Figure 2.17: The repeating module in an LSTM.

The core idea behind LSTMs are the cell state and its different gates; the key components enabling past information to be used at a later time, thus addressing the vanishing gradient problem. On the one hand, the cell state (also known as the long-term memory), is the memory of the cell. It can carry relevant information during the processing of the sequence, including information from the earlier time steps. A good analogy for a better understanding, is to see the cell state as a conveyor belt which runs in parallel to the sequence being processed, where information from the sequence can be put on the conveyor belt at any point, be transported to a later time step, and be taken when it is required.

On the other hand, LSTMs have the ability of removing or adding information to the cell state. It is carefully regulated by means of internal mechanisms called gates. Gates are composed of a neural network layer with *sigmoid* activation function and a pointwise multiplication operation, and they decide what information to remove, maintain, or add from the cell state. As such, these gates have weights that are learned during the training procedure. Specifically, there are three gates within an LSTM cell: the forget gate, the input gate, and the output gate. Figure 2.18 shows how an LSTM unit is structured internally.

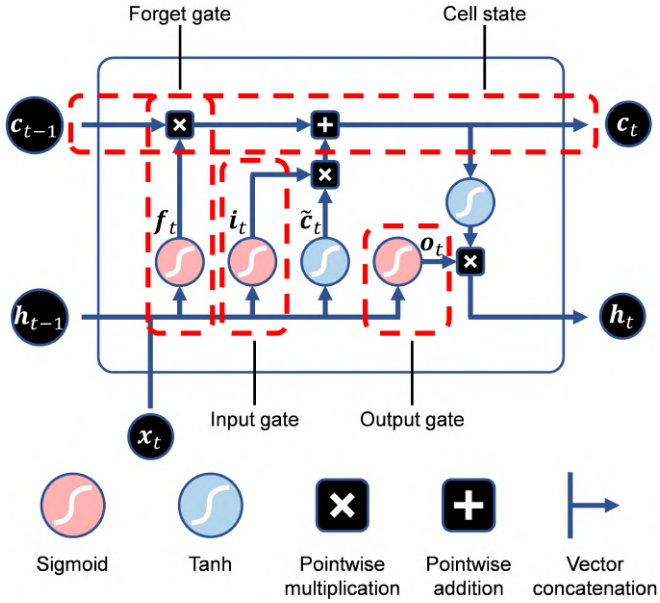


Figure 2.18: The internal structure of an LSTM cell.

The three cell inputs are the input data vector at the current time step ($\boldsymbol{x}_t$), the hidden state of the previous time step (the previous output of the cell) ($\boldsymbol{h}_{(t-1)}$), and the state of the cell (the memory) ($\boldsymbol{c}_{(t-1)}$), whereas the two cell outputs are the current hidden state (the current output of the cell) ($\boldsymbol{h}_t$), and the updated memory of the cell ($\boldsymbol{c}_t$). Regarding the gates, they will be explained below in an operational step by step to have a global comprehension of how an LSTM cell works:

1. **Forget gate ($\boldsymbol{f}_t$)**: the first step is to decide what information to throw away from the cell state. It is done by means of $\boldsymbol{f}_t$ by passing the information of $\boldsymbol{h}_{(t-1)}$ combined with $\boldsymbol{x}_t$ through its neural network layer with *sigmoid* activation function, giving values between 0 and 1 to the cell state (1 representing keeping all the information while 0 representing removing all the information). Mathematically, this gate can be expressed as follows:

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{U}_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \tag{2.8}$$

being $\boldsymbol{W}_f$, $\boldsymbol{U}_f$, and $\boldsymbol{b}_f$ the weights and biases of the forget gate.

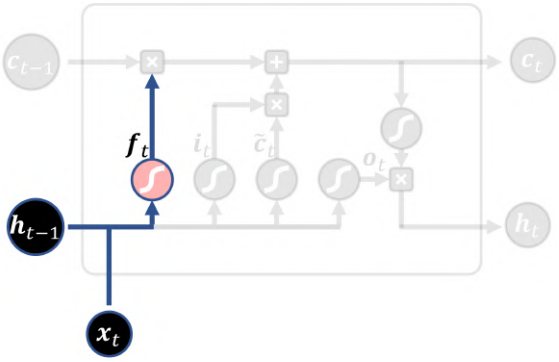


Figure 2.19: Forget gate operations.

2. **Input gate ($\boldsymbol{i}_t$)**: the second step is to decide what information from the input to update to the cell state. It is done in two parts: on the one hand, and in the same way as for $\boldsymbol{f}_t$, by means of $\boldsymbol{i}_t$ by passing the information of $\boldsymbol{h}_{(t-1)}$ combined with $\boldsymbol{x}_t$ through its neural network layer with *sigmoid* activation function, and on the other hand, by means of a vector of new state candidates ($\tilde{\boldsymbol{c}}_t$) by passing the same combined information through an hyperbolic tangent activation function (*tanh*), giving values between -1 and 1 to its output. $\tilde{\boldsymbol{c}}_t$ gives rise to the candidates for the memory of the cell, while $\boldsymbol{i}_t$ decides which values are updated. In the next step the pointwise multiplication of these values will be used to calculate the cell state. These two operations are described mathematically as:

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i \boldsymbol{x}_t + \boldsymbol{U}_i \boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \quad (2.9a) \qquad \tilde{\boldsymbol{c}}_t = tanh(\boldsymbol{W}_c \boldsymbol{x}_t + \boldsymbol{U}_c \boldsymbol{h}_{t-1} + \boldsymbol{b}_c) \quad (2.9b)$$

being $\boldsymbol{W}_i$, $\boldsymbol{U}_i$, $\boldsymbol{b}_i$, $\boldsymbol{W}_c$, $\boldsymbol{U}_c$, and $\boldsymbol{b}_c$ the weights and biases of the input gate, and the state candidates respectively.

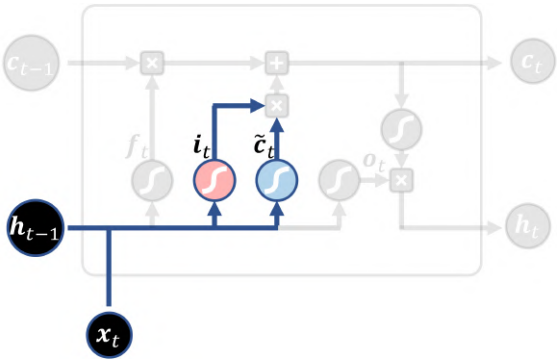


Figure 2.20: Input gate operations.

3. **Cell state ($\boldsymbol{c}_t$)**: the third step is to update the previous cell state. It is done by means of the sum of the pointwise multiplications of $\boldsymbol{c}_{t-1}$ by $\boldsymbol{f}_t$, and $\tilde{\boldsymbol{c}}_t$ by $\boldsymbol{i}_t$. The first product

removes or maintains information, whereas the second product adds (or not) information. In mathematical terms, this computation is expressed in the following way:

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t \tag{2.10}$$

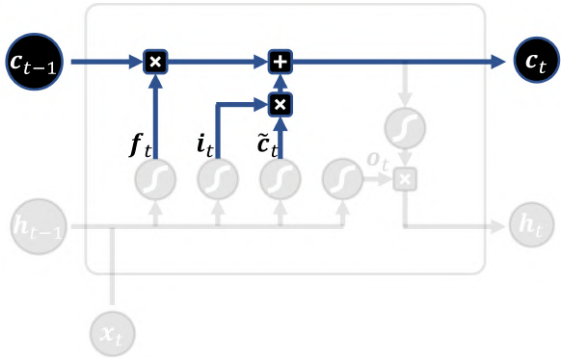


Figure 2.21: Calculating cell state.

4. **Output gate (o_t)**: the last step is to decide what to output based on input and the cell state. It is done in two parts: on the one hand, by means of $\boldsymbol{o}_t$ by passing the information of $\boldsymbol{h}_{t-1}$ combined with $\boldsymbol{x}_t$ through its neural network layer with *sigmoid* activation function, and on the other hand, by passing $\boldsymbol{c}_t$ through a *tanh* activation function. The output of the *tanh* gives rise to the $\boldsymbol{c}_t$ values between -1 and 1, while $\boldsymbol{o}_t$ decides which values are output. The pointwise multiplication of these values is $\boldsymbol{h}_t$. It is described as follows:

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o \boldsymbol{x}_t + \boldsymbol{U}_o \boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \tag{2.11a}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot tanh(\boldsymbol{c}_t) \tag{2.11b}$$

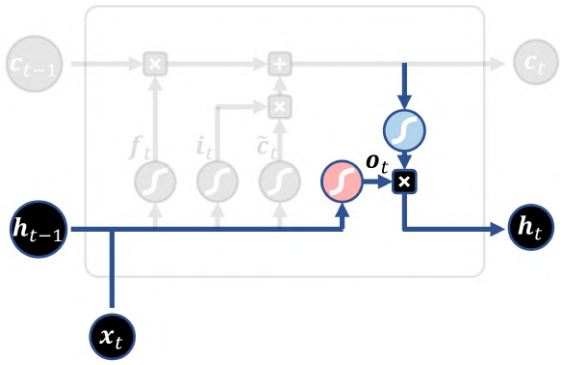


Figure 2.22: Output gate operations.

As could be noted, the *sigmoid* and *tanh* activation functions play an important role in this mechanism. The first one keeps or forgets information depending on its output values (between 0 and 1), and the second one keeps the information between a workable range of values (between -1 and 1).

Chapter Summary

Reached this point, it is assumed that the theoretical background of this project has been dealt. On the one hand, the working scenario has been presented from a general view, such as the plant design, to a more concrete and interesting view for this study; that of the controllers. On the other hand, the controller structure which will replace the ones of the working scenario has been introduced. Finally, the basics about the key components of the proposed controllers, i.e., the ANNs, was covered deepening in the LSTM networks and their functioning. In the next chapter, the general procedure used to design neural network models as the LSTMs models of this project, will be presented.

Chapter 3

Workflow for the Design of Neural Network Models

Creating a neural network model, as well as any other machine learning model, involves an iterative process of development and refinement, depicted in Figure 3.1.

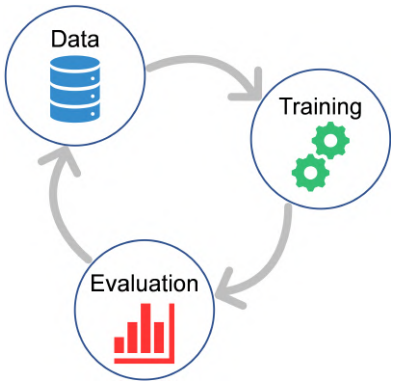


Figure 3.1: Workflow diagram for the creation of a neural network model.

In summary, once the data is collected, studied, and processed, the model is trained to transform the inputted data into the desired results by showing it representative samples. After being trained, it is tested with data it has not seen before and evaluated to know how it works. If the model is performing well enough, it is ready to operate, otherwise the process would be repeated, or the algorithm would be directly fit again with a different configuration.

3.1 Data Gathering, Analysis, and Preprocessing

Get clean data is the first step in designing any data-based model and it is critical because the quantity and quality of the information directly affects the performance of the model.

In order to obtain the datasets, two years of influent considering dry, rainy, and stormy weathers are used to generate the data by means of the BSM1 simulation scenario in open loop. Gather information with this setup entails: remove the default controller to be replaced from the original simulation benchmark, create a random square pulse signal (the manipulated variable) with reasonable characteristics (similar to those specified in the BSM1 description in therms of minimum and maximum of the signal, and pulse length), and finally introduce this signal filtered into the BSM1 to pick up its response.

A previous consideration to the introduction of the pulse signal to the plant, is the filtering of this signal with the aim of employing smooth variations, pretending a more realistic situation. Figure 3.2 outlines an example of this process, which is nothing more than a kind of ideal low-pass filter since it eliminates all frequencies above a cutoff frequency while passing those below unchanged.

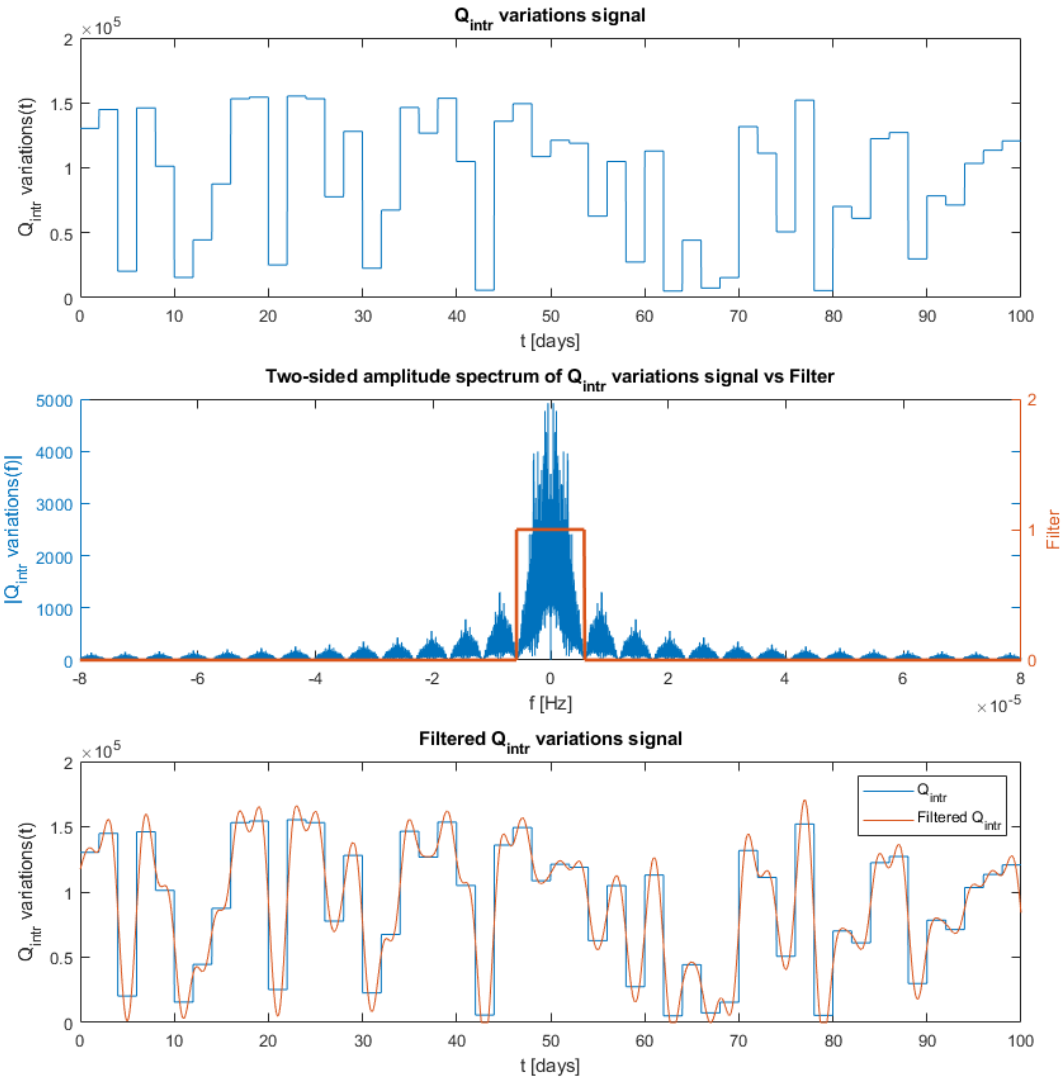


Figure 3.2: Filtering of the internal recirculation flow rate (Q_{intr}) signal.

The random square pulse signal (first 100 days shown in Figure 3.2 top) is passed into the frequency domain by means of the fast Fourier transform (FFT), and as expected, the FFT of this nature gives rise to a kind of *sinc* function (shown in Figure 3.2 middle). Then, the transform is multiplied by a rectangular function letting pass, for instance, the frequencies included in its first lobe. As consequence of this operation, when going back to the time domain, the signal shows that the transitions between values are not immediate but have a certain slope (first 100 days shown in Figure 3.2 bottom).

With the manipulated variable filtered, it is introduced into the pertinent process to take the necessary signals in its response and be able to model it. Initially the raw data has the form shown in the example below.

	time	SI	SS	XI	XS	...	SALK	TSS	Qi	KLa	SO_out
0	0.000000	6.651196	1.319671	424.452958	15.941808		1.004701	815.749119	67299.266655	262.858491	7.072919
1	0.010417	6.407897	1.450425	363.565933	19.435902		1.361027	875.597305	61897.067944	274.130976	7.448862
2	0.020833	7.248697	1.658750	325.310177	25.100516	...	1.571280	926.009554	64218.537931	274.130951	7.312797
3	0.031250	8.724000	1.859670	319.067935	31.302774		1.742632	996.468206	66449.491433	222.078550	6.760477
4	0.041667	10.493027	1.994099	335.885783	36.641204		1.916414	1077.518721	62526.794820	131.163506	5.629228

Figure 3.3: Head of the initial dataframe for the ANN_{dir} structure of the dissolved oxygen in the fifth thank ($S_{O,5}$).

Each row of the previous table represents a different time step of the two years compiled (for the sake of brevity, sampling intervals of 15 minutes are equivalent to 1/96 seconds in the simulator), while each of the first columns corresponds to the different concentrations available to be taken as input variables to train the neural network model together with the last column, belonging to the concentration taken as output, i.e., the target, the one to be predicted.

At this point, there is an interest to know the relationship between the output variable of the process to be controlled ($S_{O,5}$ and $S_{NO,2}$ for the first and second controllers respectively) and the available features. That is why it is important to analyze the data: to determine how it should be used. To that end, different measures like the correlation and mutual information (MI), chosen in this work, can be used as guiding tools. As such, these measures are not computed to know exactly which variables to use in the training of the neural network models (it is an iterative process), but to have a reference on which variable can favor their functioning.

Correlation is a measure that quantifies the linear association between a pair of variables, i.e., how strongly one variable depends on the other [Bon17]. There are many measures for it, but in this case, the Pearson product-moment correlation coefficient (PPMCC) is used to find the pairwise correlation of the different datasets. The data analysis and preprocessing phase is carried out using Python as it was indicated in the organization of the thesis. Specifically, the correlation has been calculated by means of the $pandas.DataFrame.corr()$ function, as illustrated in the exemplary matrix of Figure 3.4.

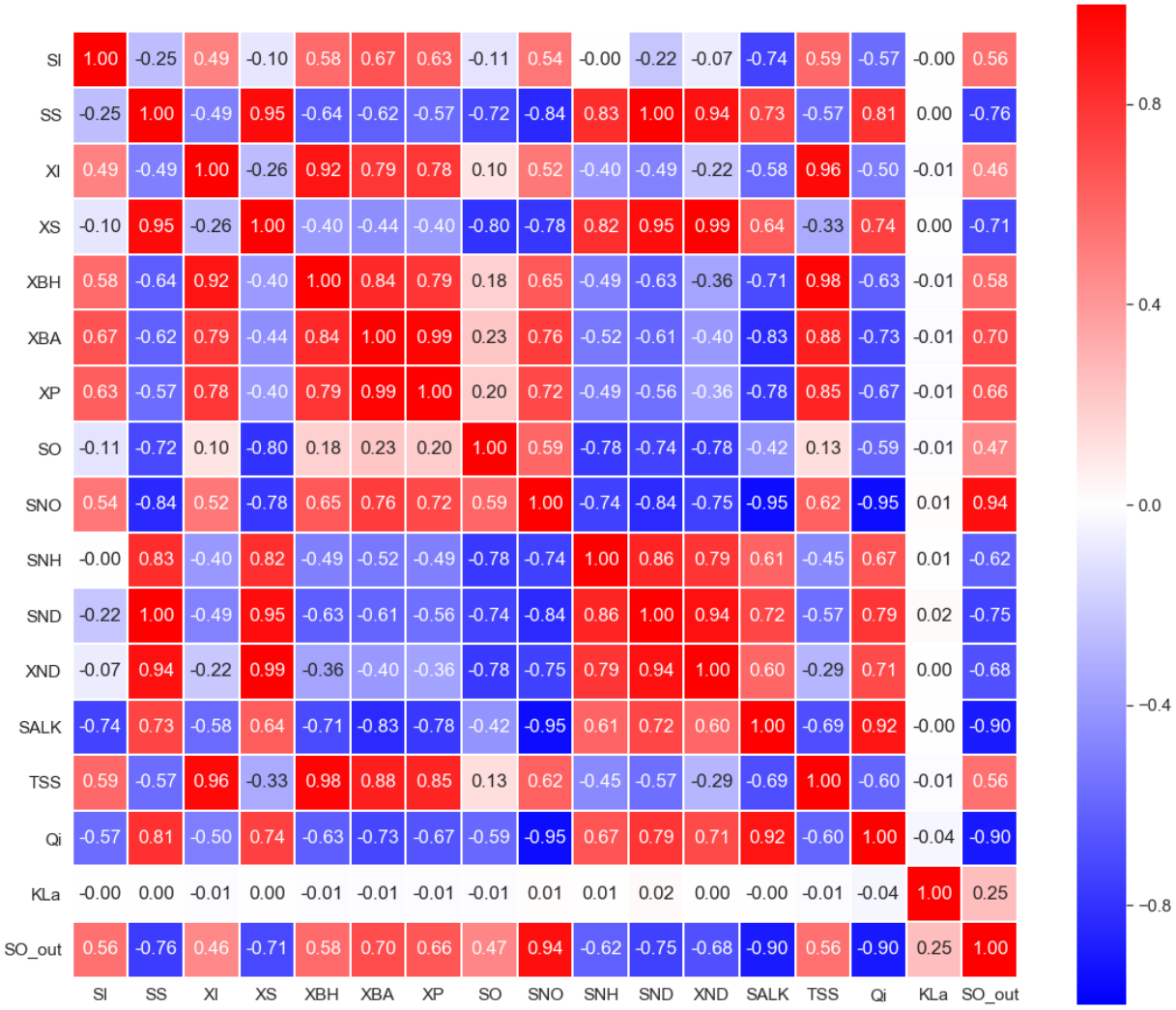


Figure 3.4: Correlation matrix of the data from the entrance to the fifth tank, its oxygen transfer coefficient ($K_L a_5$), and the dissolved oxygen at the exit of said fifth tank ($S_{O,5}$).

This table format summarizes the correlation between all possible combinations of variables with each cell. As can be noticed, the matrix is square since it has the same components in the rows and in the columns. The line of 1.00s going from the top left to the bottom right is the main diagonal, which shows that each variable always perfectly correlates with itself. It is also possible to observe that the matrix is symmetrical; the same correlation is shown above the main diagonal being a mirror image of those below the main diagonal.

In terms of the correlation coefficient, it can take a range of values from 1.00 to -1.00, and each one can be easily interpreted through its two key components:

- **Magnitude**: the larger the magnitude (closer to 1.00 or -1.00), the stronger the correlation. A value of 0 indicates that there is no association between the two variables.
- **Sign**: if positive, there is a positive association; that is, as the value of one variable

increases, so does the value of the other variable. If negative, there is a negative association; that is, as the value of one variable increases, the value of the other variable decreases.

To see this visually, the scatterplots of two pairs of variables with high correlation (positive and negative respectively) are represented in Figure 3.5.

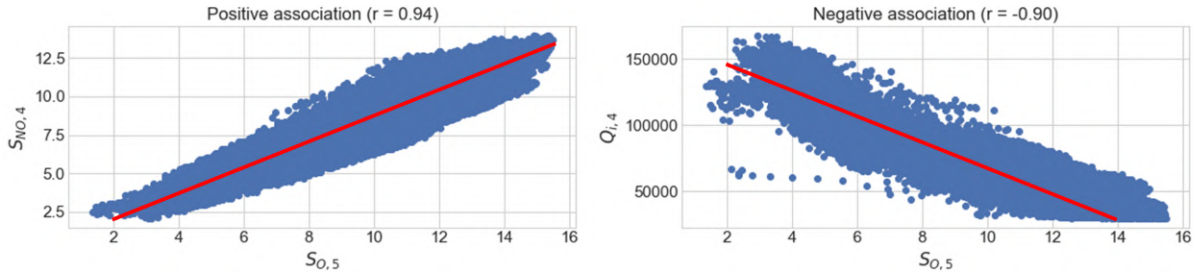


Figure 3.5: Scatterplots of the dissolved oxygen at the exit of the fifth tank vs the nitrate (left) and the flow rate at the entrance of said fifth tank (right).

If a line of best fit (a straight line that best represents the data on the scatterplot) is drawn through the data of the two pairs of variables (indicated in red), it can be observed that the sign of the correlation coefficient matches the slope of the line. The first subplot (r = 0.94) clearly shows that with an increase in the $S_{O,5}$, there seems to be an increase in the $S_{NO,4}$. On the other hand, the second subplot (r = -0.94) exhibits a decrease in the $Q_{i,4}$ as the $S_{O,5}$ increases.

The correlation matrix in Figure 3.4 is enough for the case of the first control loop, since as will be seen in Section 4.1, data has only been taken from the input of the fifth tank to model the process of that controller. However, for the case of the second control loop, and because the process to be modeled is more complex, data has been taken from the input and output of the first tank, the output of the first combiner, and the output of the settler as will be seen in Section 4.2. So, a total of four correlation matrices have been computed considering these points along with the influent flow rate (Q_{in}), the internal recirculation flow rate (Q_{intr}), and the nitrate of the second tank ($S_{NO,2}$).

A tool like the Pearson's correlation refers to the degree to which a relationship is linear. In this way, it will not be useful for determining the relationship between two variables with a non-linear relationship. Instead, it might be better described by another statistical measure like the MI.

MI is a measure that quantifies the mutual dependence between a pair of variables, i.e., how much information is obtained about one random variable through observing the other random variable [Cov12]. As discussed above and in contrast to correlation, MI is not limited to linear dependence. In order to calculate it, the formula of the MI between two random variables X and Y has been taken as a starting point:

$$I(X;Y) = H(X) - H(X|Y) \tag{3.1}$$

being $H(X)$ the entropy for X, which is the uncertainty of a single variable, and $H(X|Y)$ the conditional entropy, which is the entropy of the random variable X conditional on the knowledge of the random variable Y. In the same way as for the previous measure, an exemplary matrix containing this kind of computations is shown in Figure 3.6.

Figure 3.6: Mutual information matrix of the data from the entrance to the fifth tank, its oxygen transfer coefficient ($K_L a_5$), and the dissolved oxygen at the exit of said fifth tank ($S_{O,5}$).

For this matrix, each cell represents the mutual information between two variables, and it also maintains the square and symmetrical shape as in the previous instance by having the same components in rows and columns. The main diagonal, according to the self-information, generally exhibits the highest values as well, 1.00 being the largest due to the normalization of the matrix. MI is always larger than or equal to 0.00, where a large value indicates a great

relationship between the two variables; a low value indicates a small relationship; and 0.00 means the variables are independent.

Compared to the matrix of Figure 3.4, it can be appreciated that for the last row (the one of interest for reporting the results related to the controlled variable), the highest correlation values mostly coincide with the highest MI values. However, some concentration such as $S_{NH,4}$ shows a higher priority according to MI (it is the 5^{th} variable with the best MI being the 10^{th} with the strongest correlation), and that is why both measures have been taken into account as reference in the selection of variables to train the network; because they are complementary and describe different aspects of association.

Another contrast to the correlation matrix is that the results of the main diagonal are not always maximum. This fact can be easily explained by plotting a variable with large self-information, and a variable with low self-information as in Figure 3.7.

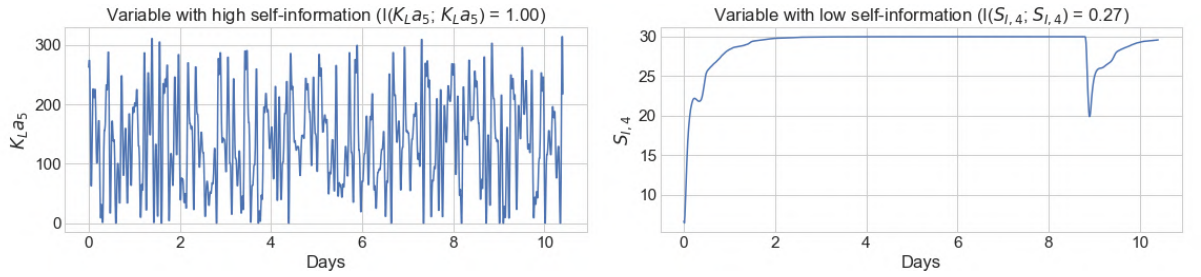
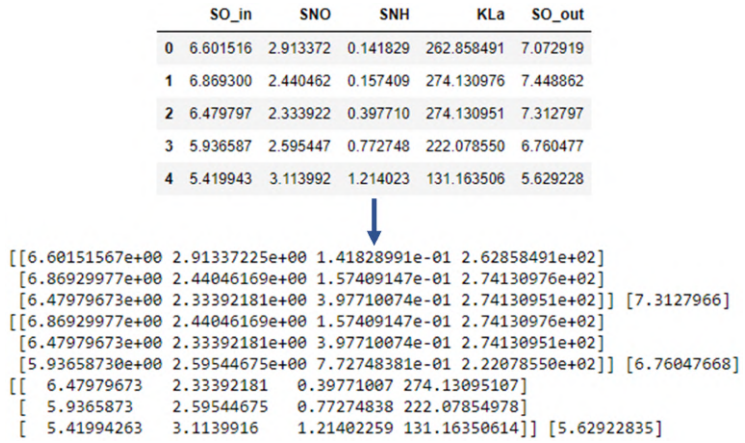


Figure 3.7: Plots of the first days of data of the oxygen transfer coefficient of the fifth tank (K_La_5) (left), and the soluble inert organic matter of the fourth tank ($S_{I,4}$) (right).

First, it should be noted that the self-information of a variable corresponds to the entropy of that variable since the conditional entropy of the same variable is zero. Then, given that the entropy measures the uncertainty of an information source, or in other words the amount of average information contained in the symbols (values) used, when all symbols are equally likely (as in the case of the random variable K_La_5 generated), they all provide relevant information and entropy (or self-information) is maximum. Otherwise, when certain symbols are more likely than others (as it is the case of 30 for the $S_{I,4}$ concentration), the uncertainty (or self-information) is lower.

Having an idea of which signals to use in the model training, it just remains to apply the necessary preprocessing techniques to have the desired dataframe. Particularly for this project, data organization is necessary, because as outlined in Chapter 2, LSTM models learn a function that maps a sequence of past observations as input to an output observation. Given that the selected data is of the type shown in Figure 3.3 (multivariate time series data with multiple input series), and there is a single observation per time step for each input series, data transformation must be done by means of the sliding window method. This technique divides the original sequence into multiple samples, where a certain number of time steps (the window width) is

used as input, and one time step is used as output. For a better understanding, Figure 3.8 presents an example of the effect of applying the sliding window (with a window width of three time steps) to the concentrations used in the first controller.



	SO_in	SNO	SNH	KLa	SO_out
0	6.601516	2.913372	0.141829	262.858491	7.072919
1	6.869300	2.440462	0.157409	274.130976	7.448862
2	6.479797	2.333922	0.397710	274.130951	7.312797
3	5.936587	2.595447	0.772748	222.078550	6.760477
4	5.419943	3.113992	1.214023	131.163506	5.629228

```
[[6.60151567e+00 2.91337225e+00 1.41828991e-01 2.62858491e+02]
 [6.86929977e+00 2.44046169e+00 1.57409147e-01 2.74130976e+02]
 [6.47979673e+00 2.33392181e+00 3.97710074e-01 2.74130951e+02]] [7.3127966]
[[6.86929977e+00 2.44046169e+00 1.57409147e-01 2.74130976e+02]
 [6.47979673e+00 2.33392181e+00 3.97710074e-01 2.74130951e+02]
 [5.93658730e+00 2.59544675e+00 7.72748381e-01 2.22078550e+02]] [6.76047668]
[[  6.47979673   2.33392181   0.39771007 274.13095107]
 [  5.9365873    2.59544675   0.77274838 222.07854978]
 [  5.41994263   3.1139916    1.21402259 131.16350614]] [5.62922835]
```

Figure 3.8: Head of the selected dataframe for the ANN_{dir} structure of the dissolved oxygen in the fifth thank ($S_{O,5}$) (top) and its transformation via the sliding window (bottom).

The result of this example leads to samples with three time steps of each parallel series as input associated with the value of the output series at the third time step. It means that the dimension of the input component has a three-dimensional structure: the first dimension is the number of samples (three included in Figure 3.8 bottom), the second dimension is the window width (three chosen for this instance), and the third dimension is the number of features (four in this case). With this configuration, LSTM models have enough information to learn a mapping from an input sequence to an output value.

Finally, the feature standardization has been required to decrease the heterogeneity in the data making the values of each feature have zero-mean and unit-variance, thanks to the *sklearn.preprocessing.StandardScaler*() function.

3.2 Model Training

At this second step, proper inputs and output are known. However, there is still a last unknown concerning the mathematical function to transform these inputs into the desired goal. So, the next step is to expose the learning algorithm to reference samples in order to build (train) an accurate model in mapping the input to output values.

Between the several possibilities to make this partition, in this project, the K-fold cross-validation is adopted [Ber18]. For cases with limited or unbalanced data samples, the performance obtained for one validation set can be very different to the performance obtained for a different validation set. K-fold cross-validation provides a solution to this issue by dividing the

available data for training and validation into folds of equal size and ensuring that each fold is used as validation set at the different points of the available data. Figure 3.9 clearly describes this technique with the sample quantities used in this work.

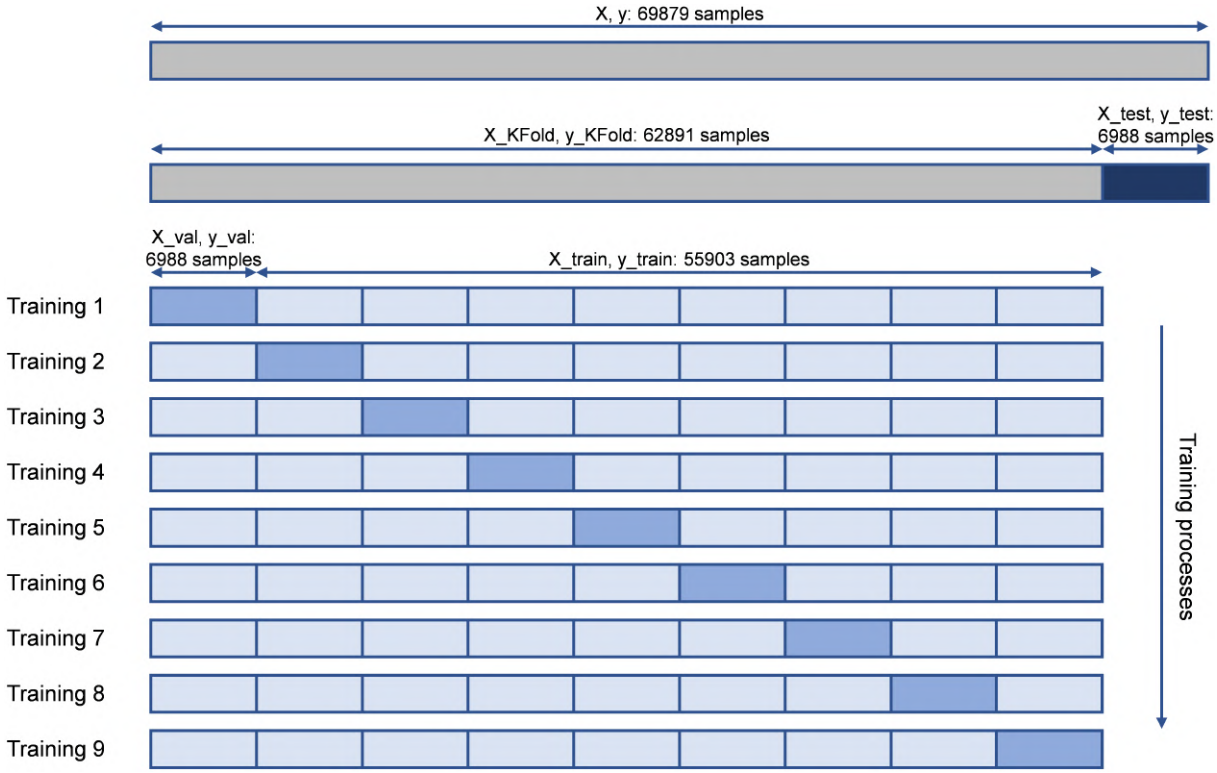


Figure 3.9: Application of Nine-fold cross-validation.

From the whole dataset (69879 samples), first a part is reserved for testing purposes (6988 samples, 10% of all data), and then, K-fold cross-validation is applied to the remaining part (62891 samples, 90% of all data). With the implementation of this tool, the part devoted to training and validation is split into K partitions of equal size (nine of 6988 samples, 10% of all data), and K trainings are done evaluating a different partition in each one.

Consequently, K different models are obtained from this training, each of them giving a certain score used to assess overfitting. To get the final model performance, the average of the K scores is computed. Finally, the model used to operate can be anyone of the K produced.

3.3 Model Evaluation

The third and final step consists in evaluating the model, an essential part of building an effective machine learning or deep learning model, as it allows for a critical feedback from metrics and make improvements. This step determines when the development cycle ends: when the metrics in the model evaluation show the desired operation.

In this way, model evaluation allows to check the model performance against unseen data (data that has never been used for training), get an idea of how the model will work in the real world and therefore determine if the model is good or not.

Model evaluation metrics are required to quantify model performance. When someone says something is good (whether it is a car, a country, or a machine learning model), the first question is on what basis is this statement being made (e.g. consumption for the case of the car, economic status for the case of the country, and accuracy for the case of the machine learning model). For machine learning models, the choice of evaluation metrics depends on each situation as each of them can address different problems (such as classification or regression problems) under different conditions, and that is why it is important to understand the context first.

The performance of the ANNs of this work will be evaluated accordingly to five metrics: the mean absolute percentage error (MAPE), the mean arctangent absolute percentage error (MAAPE), the root mean squared error (RMSE), the normalized root mean squared error (NRMSE), and the coefficient of determination (R^2).

- **MAPE**: is the mean or average of the absolute percentage errors of forecasts. It is scale-independent and easy to interpret because it provides the percentage of error in the prediction. Consequently, the smaller the MAPE the better the forecast. It is defined as:

$$MAPE = \frac{1}{N} \cdot \sum_{i=1}^{N} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \cdot 100 \tag{3.2}$$

  where N corresponds to the number of examples, y_i corresponds to the i^{th} sample of the real output data, and $\hat{y}_i$ is the i^{th} predicted value.

  However, the MAPE produces infinite or undefined values (outliers) for zero or close-to-zero actual values, and for this reason the MAAPE is also proposed.

- **MAAPE**: is the mean or average of the arctangent absolute percentage errors of forecasts [Kim16]. The bounded range of the arctangent function provides a maximum AAPE of $\pi/2$. As MAPE, good predictions have results close to zero. It is computed as follows:

$$MAAPE = \frac{1}{N} \cdot \sum_{i=1}^{N} arctan\left(\left| \frac{y_i - \hat{y}_i}{y_i} \right| \right) \cdot 100 \tag{3.3}$$

- **RMSE**: is a globally known metric, frequently used to measure the differences between values. It is a scale-dependent measure with desirable values around zero. Its formula is the next one:

$$RMSE = \sqrt{\frac{1}{N} \cdot \sum_{i=1}^{N} (y_i - \hat{y}_i)^2} \tag{3.4}$$

Nevertheless, the RMSE is scale dependent. Since the scales of the real output data differ across the different ANNs, the NRMSE is also considered to compare the results of the different ANNs with similar conditions.

- **NRMSE**: is the mean of the RMSE. Obviously, values near zero are interesting. In this situation, the RMSE is normalized by the difference between maximum and minimum, giving rise to the following equation:

$$NRMSE = \frac{\sqrt{\frac{1}{N} \cdot \sum_{i=1}^{N} (y_i - \hat{y}_i)^2}}{y_{max} - y_{min}} \tag{3.5}$$

y_{max} and y_{min} being respectively the maximum and minimum values of the real output data.

- R^2: measures the amount of the data variance that is explained by the model and ranges from 0 to 1. Thus, a result of 1 reveals a perfect correlation between values [Pis19c]. R^2 can be calculated as:

$$R^2 = \frac{\left(\sum_{i=1}^{N} \left(\hat{y}_i - \overline{\hat{y}}\right) \cdot (y_i - \overline{y})\right)^2}{\sum_{i=1}^{N} \left(\hat{y}_i - \overline{\hat{y}}\right)^2 \cdot \sum_{i=1}^{N} (y_i - \overline{y})^2} \tag{3.6}$$

where $\overline{\hat{y}}$ corresponds to the mean of the predicted values, and the mean of the real values corresponds to $\overline{y}$.

Chapter Summary

As summary, this chapter has presented a vision of the design process of deep learning models, more oriented to the one followed in the elaboration of the proposed controllers (adapted with examples from the project itself) which will be seen in the next chapter. This is why the workflow was explained using data of the BSM1 itself. First, data gathering, analysis, and preprocessing has been presented. It is a very important part of the development of neural networks, which sometimes remains in the background and involves a high workload. Finally, the procedure that will be used to train and evaluate the ANNs of the IMCs has been shown. Thus, in the same way that in Chapter 2 the metrics to evaluate the controllers were defined, the metrics to evaluate the neural networks have been specified in the last section of this chapter.

Chapter 4

ANN-IMC Strategy in WWTPs

4.1 ANN-IMC Control of the Dissolved Oxygen of the Fifth Reactor Tank ($S_{O,5}$)

In this section, an IMC controller based on ANNs is designed to control the dissolved oxygen of the fifth reactor tank ($S_{O,5}$) of the BSM1. The core idea is to improve the performance of the original PI controller dedicated to this task. Consequently, the goal of this episode is to complete the schematic shown in Figure 4.1.

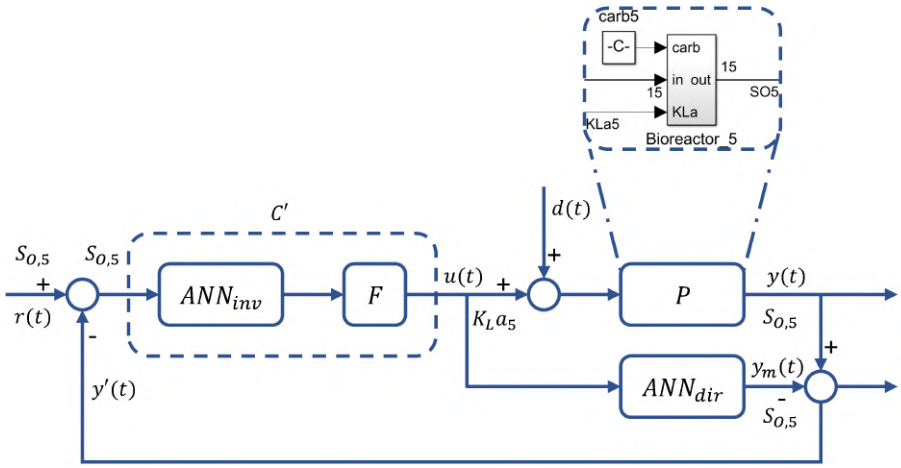


Figure 4.1: ANN-IMC controller structure for $S_{O,5}$.

As it is observed, in this case the real process P is basically the fifth bioreactor tank. The filter F consists in a first-order filter, where the gain will be fixed to 1 and the time constant will be selected based on the results of the control metrics. The controlled variable, associated with $r(t)$, will be the $S_{O,5}$ set point, while the actuation variable, corresponding to $u(t)$, will be the K_La_5 concentration. In such a way, the controllers track the $S_{O,5}$ variations modifying the

K_La_5 coefficient. Finally, the work of this section is summarized in finding the missing parts of the controller structure, i.e., the ANN_{dir} and the ANN_{inv}, in order to achieve a model output signal $y_m(t)$ as similar as possible to the process output signal $y(t)$. In other words, to minimize the effects of process-model mismatch.

4.1.1 Direct and Inverse ANNs

In line with the workflow for the design of ANNs models, the first step is to get data, analyze them and preprocess them. For that, two years of influent considering dry, rainy, and stormy weathers is used together with a filtered two-years K_La_5 variable. This variable consists in normally distributed random numbers around 0 and 300 with variations of 1 hour taking into account the characteristics of this concentration under the description of the benchmark, to collect data of the BSM1 response in open loop. Using a manipulated variable with a range of random values to collect the data allows the ANN to have plenty of examples when training, so that it can generalize when it receives different values as input.

For this first controller, a study of correlation and mutual information (MI) of the concentrations collected at the entrance of the fifth reactor tank, along with K_La_5 and $S_{O,5}$, was carried out as depicted in Figures 3.4 and 3.6. Some of the signals that exhibit higher values of these two measures employed for the information study are the $S_{O,4}$, the $S_{NO,4}$, and the $S_{NH,4}$ [Pis19a]. They are commonly used to train both, the ANN_{dir} and the ANN_{inv}. Then, for the ANN_{dir} in particular, the K_La_5 coefficient must be compulsorily employed because this model represents the process being controlled, and as seen in the description of BSM1, the dissolved oxygen control loop takes the K_La_5 to control the $S_{O,5}$. The rest of variables, in this case the three mentioned above, are chosen arbitrarily in terms of how many and which ones. The goal is to help the ANN_{dir} in the modeling task of the process under control with the greatest precision. Similarly, for the ANN_{inv} apart from the three concentrations shared by both processes, the essential input and output variables are exchanged, i.e., K_La_5 for $S_{O,5}$. An additional feature for the modeling of this process is that the output value of the previous instant is also used, namely $K_La_5(t-1)$. The use of this last input is based on the philosophy of nonlinear autoregressive models with exogenous inputs (NARX models), which are networks that have feedback from the output neuron [Sie97]. This feature gives a lot of information to the network since the output value of the previous instant is being passed as input at every moment. Therefore, the network will give a lot of importance to this input for predictions, and the rest of inputs will serve to slightly modify the forecasts. However, it is necessary to take care with this configuration because it can result in overfitting problems (the model can learn a function related too closely to this delayed output feature).

Regarding the preprocessing, the sliding window is applied considering a window width of 10

time steps for the two ANNs, i.e., a window width of 2.5 hours (the sampling rate is 15 minutes). This method leads to a three-dimensional input shape (the shape required for fitting an LSTM model) of 62891 samples of 10 time steps of each parallel input series for the ANN_{dir} and the ANN_{inv}, respectly represented in Figure 4.2.

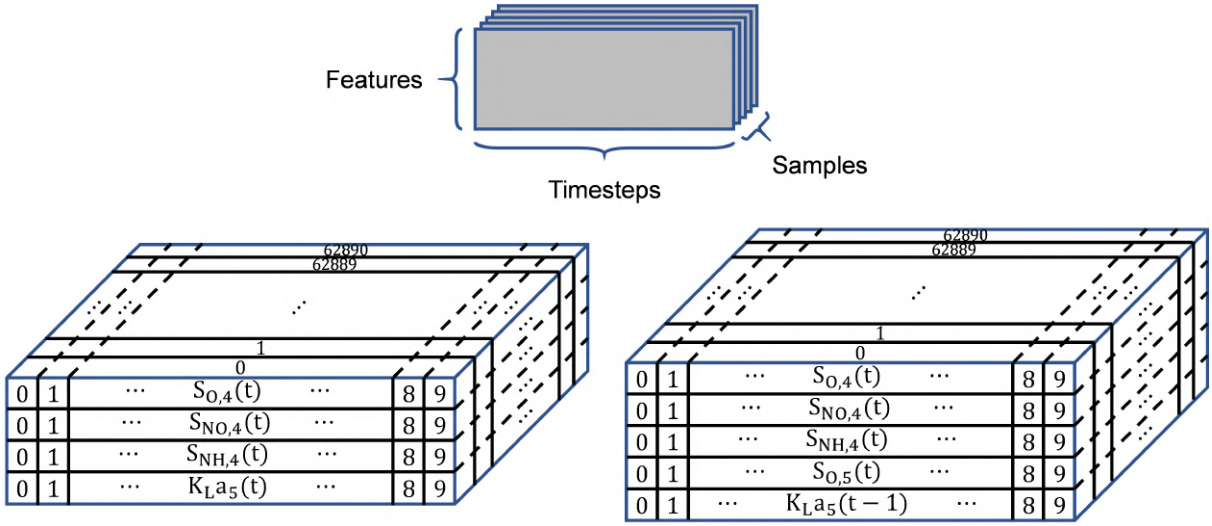


Figure 4.2: Shape of the 3D input arrays used to train the LSTM networks of the direct (left) and inverse (right) processes of the first control loop.

At this point, the ANNs can be trained, and as seen in Chapter 2, there are several algorithms for this. In this dissertation LSTM cells will be used because of their capacity in modeling time-series and time-dependent parameters such as the WWTP's influent and effluent values [Pis19b]. Their configuration was found performing a grid search, which is nothing more than a trial and error process to find the structures offering the best performance, in this case it involves training different configurations and choosing the one that works better. After the search, the settings established for both ANNs are synthesized in Table 4.1.

LSTM structures				
Model layer	**Type**	**Neurons per gate**	**Activation function**	**Regularizer**
1st	LSTM	100	*Tanh & sigmoid*	L2 penalty (0.001)
2nd	Dense	1	*Linear*	-

Table 4.1: LSTM structures for the ANN_{dir} and the ANN_{inv} of the $S_{O,5}$ ANN-IMC controller.

Each prediction structure consists of an LSTM model that has a single hidden layer of LSTM units (or neurons per gate), and an output layer used to make predictions (a.k.a. Vanilla LSTM). In this case, each model is defined with 100 LSTM units in the hidden layer and a unique neuron with a linear activation function in the output layer predicting a single numerical value for dealing with a regression problem. Thus, the difference between direct and inverse structures is in the number and type of signals used for their training. Hence, the structure

of the $S_{O,5}$ ANN-IMC controller once completed with the missing parts corresponding to the ANN_{dir} and the ANN_{inv}, is shown in Figure 4.3.

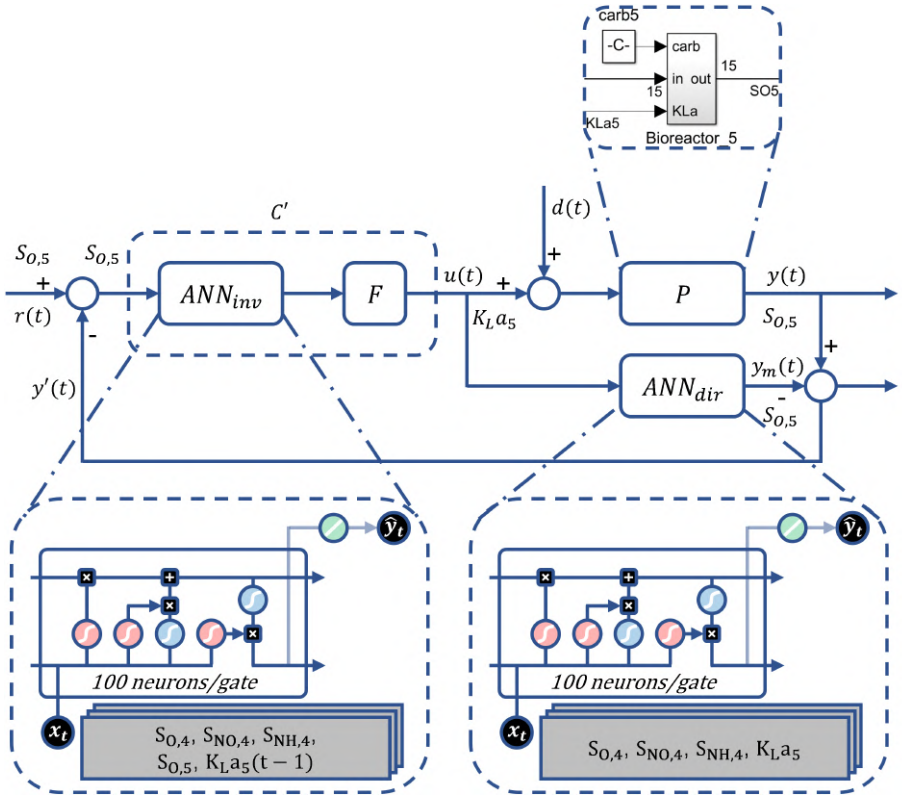


Figure 4.3: Proposed ANN-IMC controller structure for $S_{O,5}$.

In order to avoid a poor performance when making predictions on new data because of the overfitting problem discussed in Section 2.4, a regularization technique is applied to make slight modifications to the learning algorithm in such a way that the model generalizes better. Specifically, an L2 penalty of 0.001 is introduced to the loss function as a term to simplify the model. It can be understood as a kind of obstacle that is put to the learning algorithm so that it does not fit so much to the training data [Cho17].

Once the models are defined, they need to be compiled using a model optimizer and a loss function. As explained in Section 2.4, the optimizer is the search technique used to update weights in the models, and for all situations, the adaptive moment estimation (Adam) has been used for being one of the most popular gradient descent optimizer [Goo16]. The loss function is the evaluation of the model used by the optimizer to adjust the value of the weights, and also for all cases, the mean squared error (MSE) has been used for being one of the most common functions for regression problems.

The last settings regarding the model training are the number of epochs, the batch size, and the number of folds to be considered in the K-fold cross-validation. The first one is set to 200

epochs, meaning that the model will be exposed 200 times to the training dataset. The second one is set to 5000 samples, meaning that the model will see 5000 training instances before update weights. Finally, the 9-fold cross-validation shown in Figure 3.9 is performed to get an 80-10-10 split of the data into training, validation, and test, leading to 55903, 6988, and 6988 samples for each case, respectively. With all the parameters defined, all that remains is to train and evaluate these networks that will form the new $S_{O,5}$ controller, so the following lines assess their performance with the metrics defined in Chapter 3.

ANNs Performance

Once the training is done and models are built for the process of the fifth tank and its inverse, it is possible to feed them with input data similar to the one used in the training process to make forecasts. According to Chapter 3, the operation of the ANNs is analyzed by means of five metrics: the MAPE, the MAAPE, the RMSE, the NRMSE, and the R^2. Furthermore, the training method is K-fold cross-validation for K = 9 (nine experiments will be performed), and one tenth of the entire sample dataset was initially reserved for testing purposes. In average, the results of the nine models for the selected metrics gives rise to the scores of Table 4.2.

Metric	**LSTM direct model**			**LSTM inverse model**		
	Training	**Validation**	**Test**	**Training**	**Validation**	**Test**
MAPE [%]	23.212	23.552	25.114	9.544	10.055	4.388
MAAPE [%]	13.492	13.608	13.944	8.152	8.176	8.356
RMSE	0.102	0.104	0.103	2.843	2.866	2.977
NRMSE	0.015	0.016	0.016	0.009	0.009	0.009
R^2	0.997	0.996	0.997	0.999	0.999	0.999

Table 4.2: Performance of the ANN_{dir} and the ANN_{inv} of the $S_{O,5}$ ANN-IMC controller.

A model is built for each data split (fold) and is evaluated with the metrics once the prediction process is finished for the different datasets. If the data samples are unbalanced, which means that the distribution of the different types of examples among the training dataset is unequal, the nine folds would give different results. In the case of this project, the examples have been generated from data obtained from a signal with random numbers, so the entire dataset has a variety of examples and the operation of the models created for the nine divisions is very similar.

Looking at the average of the scores, it is observed that in general the nine models make good predictions, i.e., they are not underfitted. R^2 is almost one in both the direct and inverse models, being one a perfect correlation between real output and prediction. Regarding the other four metrics, they show good numbers considering that good predictions have results of these metrics close to zero. Also, it is possible to ensure that the models are not overfitted to the data of the training set because the values achieved for training and validation are quite similar. On

the contrary, if the results obtained in validation were much worse than the training, it would be an indicator of overfitting. After that, the test set provides a final check that the models are generalizing well before deploying the controller. It is important to notice that the MAPE of the LSTM inverse model for test is not a common result. However, there are two main reasons for this: the limitation of MAPE in producing large values when the actual values are close to zero [Kim16], and the signal dynamics of the test set (smoother or more variable). Although the data have been obtained with a random signal, some differences may exist along the dataset due to the randomness of the same signal, or because of the type of influent at each moment. Therefore, it is also important to emphasize the importance of using several metrics to evaluate a model. Finally, looking at the MAAPE, NRMSE, and R^2 of both models, it is observed that the inverse models generally work slightly better that the direct ones. This is because for the inverse model, the output signal of the previous instant was considered as an input, which provides a lot of information to the networks. This feature was only used to model the inverse process because the direct one gave overfitting problems.

The model parameters of one of the nine folds of each process will be exported together with the means and variances stored during the standardization procedure. Since the models of each fold offer a similar behavior, anyone is suitable to form the controller. Here, the 9^{th} and 4^{th} were chosen for the ANN_{dir} and the ANN_{inv} respectively for providing some of the best results. Figure 4.4 shows a visual example of how well these ANNs work.

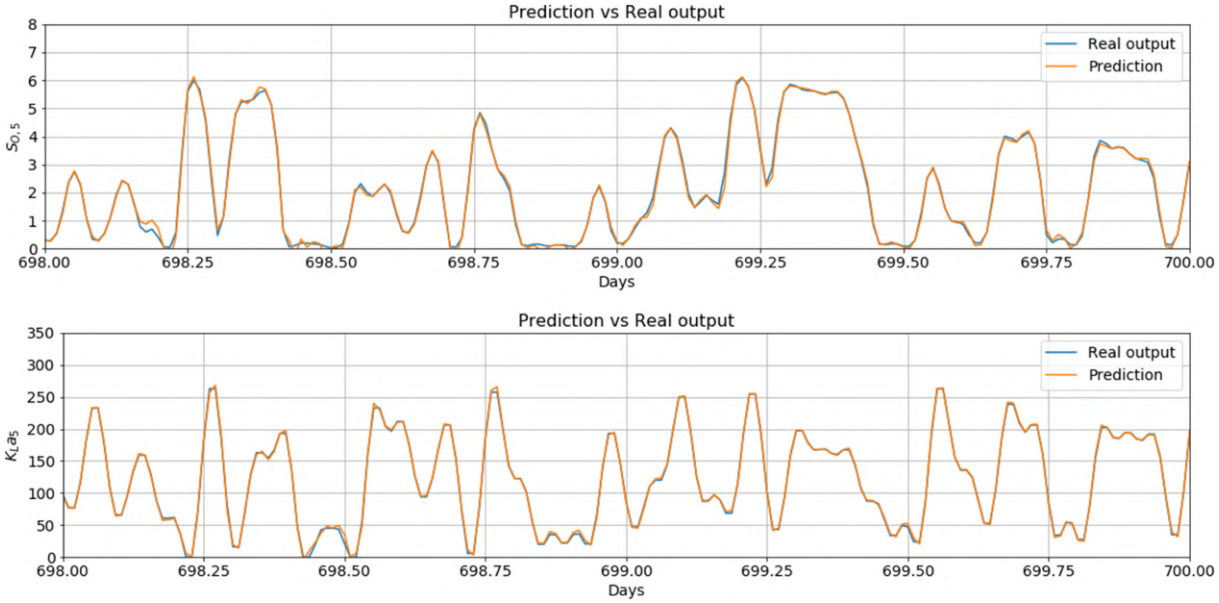


Figure 4.4: Predictions from day 698 to day 700 of the ANN_{dir} (top), and the ANN_{inv} (bottom) of the $S_{O,5}$ ANN-IMC controller.

As it is observed, a two-day prediction sample of the $S_{O,5}$ by the ANN_{dir}, and the K_La_5 by ANN_{inv} corroborates that the models offer an excellent functioning.

4.1.2 ANN-IMC Controller

Moving according to the diagram of the introduction of this thesis, it is time to implement the ANNs in MATLAB, and finally design the first ANN-IMC controller in Simulink. Starting from the beginning, once the weights and biases of the ANN_{dir} and the ANN_{inv} trained with Python are exported, they must be implemented in MATLAB and Simulink, because this will be the environment of the BSM1 where the proposed controller will be tested. In other words, the priority is to replicate the models that have been previously created in Python. To do that, both ANNs are deployed on this new platform separately, verifying that they really work as the original ones. For the sake of brevity, Figure 4.5 presents the results of this first stage only for the ANN_{dir}, since as stated above, both structures are practically identical (only the input data employed in each case changes).

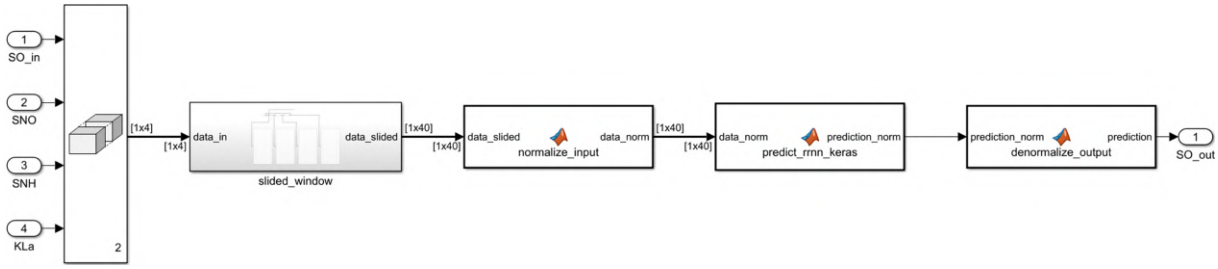


Figure 4.5: Interface layout of the ANN_{dir} structure of the $S_{O,5}$ ANN-IMC controller in Simulink.

The initial block is in charge of horizontally concatenate the four input signals to create a contiguous output signal of 1x4. These concentrations are the same as those chosen in Subsection 4.1.1 for building the ANN_{dir}. Instead, for the ANN_{inv}, five variables were used to train the network, so the output signal will be of 1x5 in that case and so on for the following boxes.

The next two blocks take care of the data preprocessing. The first of them reproduces the sliding window method by means of the schematic shown in Figure 4.6.

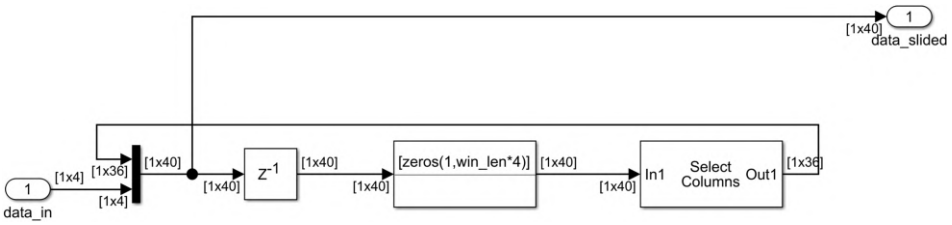


Figure 4.6: Interface layout of the sliding window of the $S_{O,5}$ ANN_{dir} structure in Simulink.

As will be seen later, the ANNs themselves will be implemented in MATLAB with simple mathematical operations such as multiplication and addition, so it is interesting to work with vectors as input rather than with three-dimensional structures as in Python. This subsystem takes 1x4 input signals and outputs 1x40 signals given that the window width was chosen as

10 time steps. The *mux* of the beginning is used to insert the new four elements (to update the window), the delay component is used as a memory to keep the previous output values, the block in the middle only initializes the signal to a vector of 1x40 zeros, and finally, the variable selector is used to drop the oldest four elements.

After that, a MATLAB function is implemented to standardize the information using the statistics of the input data saved before the training process by subtracting their mean to the slided data and dividing by their standard deviation. This action not only enables to change the values of the input features to a common scale, but also to speed up the learning and forecasting by dealing with smaller numbers.

$$x_{norm} = \frac{x - \overline{x}}{\sigma_x} \tag{4.1}$$

The next module after the preprocessing phase is the core of the ANN structures. In each of these blocks, the LSTM networks themselves are defined to carry out the predictions. For this reason, the weights and biases of the networks exported in Python are loaded to execute the same set of operations defined by Equations 2.8 to 2.11b in Section 2.4.

At the end of the architecture of these systems, the inverse function described before the prediction block is applied: the data de-standardization. In this case, this is achieved by multiplying the normalized predictions by the standard deviation of the output data and adding their mean. This operation allows the obtained forecasts to return to their real scale.

$$y = y_{norm} \cdot \sigma_y + \overline{y} \tag{4.2}$$

Having the subsystems of the $S_{O,5}$ ANN_{dir} and ANN_{inv} structures, the first controller can be formed as illustrated in Figure 4.7.

As it is shown, the controller layout follows the same structure presented in Figure 4.3. Apart from the direct and inverse LSTM models detailed above, the first-order filter can be identified in the middle as a transfer function with gain and constant time set to 1 and 1/850 respectively. Between the ANN_{inv} and this filter there is a block that does not appear in the scheme of Figure 4.3, as it simply acts as a saturator so that the output values of the ANN_{inv} are limited to a range of numbers. This unit has been added because it is also integrated in the PI controller and so the comparison between strategies is more equitable. Input and output ports 1 refer to the input and output of the process of the fifth tank. Finally, the subsystem at the bottom right oversees the calculation of the metrics that will be used to check the operation of the proposed controller which will be discussed below.

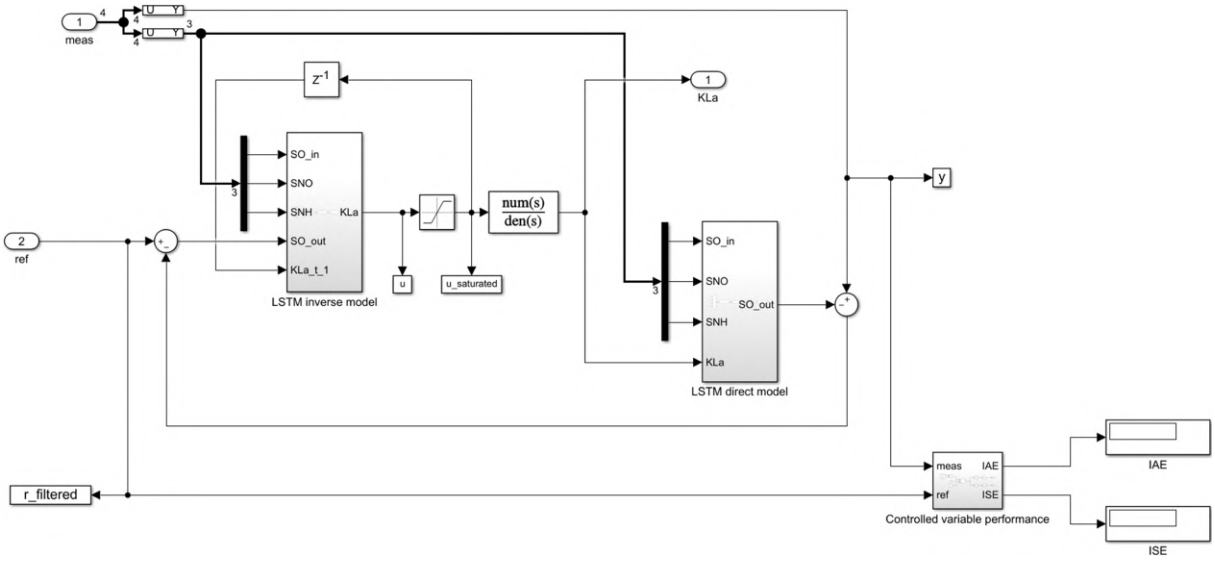


Figure 4.7: Interface layout of the $S_{O,5}$ ANN-IMC controller in Simulink.

Controller Performance

The last part of this section dedicated to the $S_{O,5}$ control loop, evaluates the operation of the default PI controller as well as the proposed IMC in order to compare both strategies. For this purpose, the same set point is used for the two structures so that the same scenario is maintained for the controllers and for the two metrics that will be employed for their examination (comparisons are fair in all aspects). This shared reference signal is generated as a random square pulse signal of 1-hour variations from 0 to 5 g/m^3 of $S_{O,5}$ since it is a range of values of this concentration that was used to train the ANNs of this controller, and consequently are numbers that the LSTM models can handle. Moreover, the performance is computed considering BSM1 influent profiles with different weathers (dry, rain and storm). As outlined in Subsection 2.2.3, the assessment of controllers is carried out in terms of the integral of the absolute error (IAE), and the integral of the squared error (ISE) criteria considering a 14 days influent but only evaluating the last 7 days. Table 4.3 provides the operation results of the default PI controller and the ANN-IMC controller of the first control loop for a $S_{O,5}$ signal with the characteristics mentioned above.

Metric	**PI**			**IMC**			**Improvement [%]**		
	Dry	**Rain**	**Storm**	**Dry**	**Rain**	**Storm**	**Dry**	**Rain**	**Storm**
IAE	1.060	1.028	1.057	0.604	0.571	0.590	43.06	44.42	44.23
ISE	0.521	0.515	0.527	0.095	0.087	0.090	81.82	83.17	82.92

Table 4.3: Performance of the PI controller and the ANN-IMC controller of the $S_{O,5}$.

The percentage decrease is calculated as an easy way to appreciate the improvements of the strategy used in this dissertation over the predetermined one established in the BSM1. These results show that the $S_{O,5}$ ANN-IMC controller offers the best performance for all three climates, in addition to giving near-zero values, and thus almost perfect tracking. For dry weather, the IMC values of IAE and ISE are correspondingly 0.604 and 0.095, which represent an improvement of 43.06% and 81.82% on the PI controller. When rainy and stormy weathers are given, these percentages of enhancement slightly rise: 44.42% and 83.17% for rain, being the case of biggest improvement, while for storm are 44.23% and 82.92%, respectively. These scores are summarized in an average enhancement of 43.90% for IAE, and 82.64% for ISE, which prove that the dissolved oxygen control strategy by default is greatly improved.

In the same way that a visual example of the operation of the ANN_{dir} and the ANN_{inv} was given in the previous subsection, Figure 4.8 demonstrates the monitoring of the $S_{O,5}$ set point for the different weathers by the proposed IMC controller.

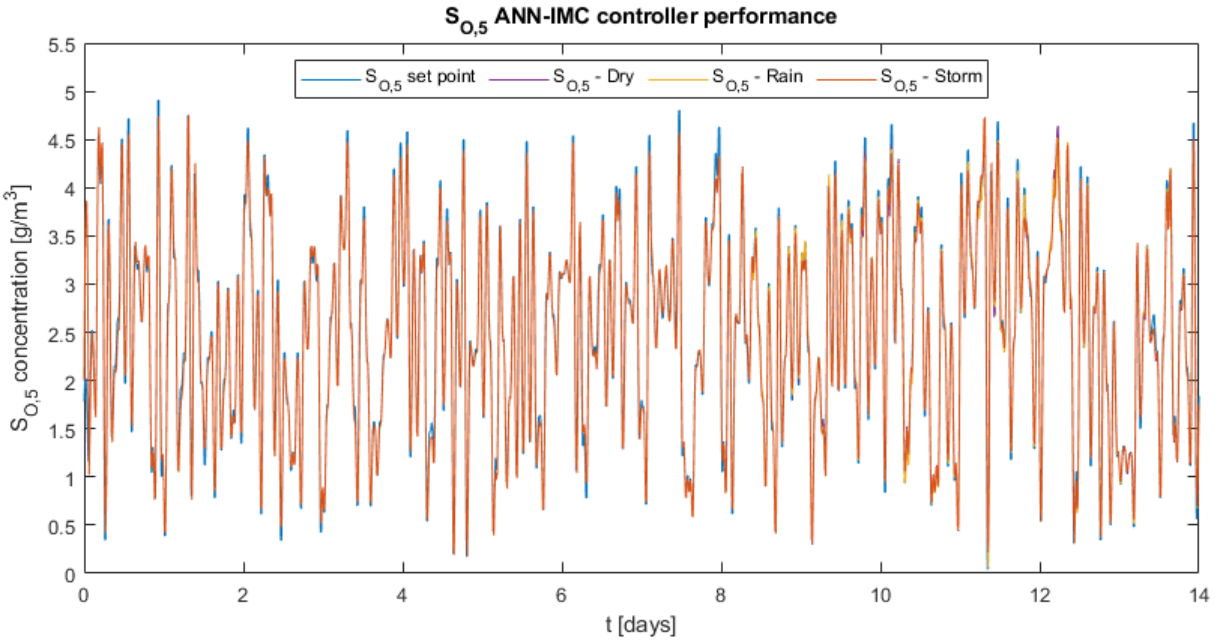


Figure 4.8: Tracking of $S_{O,5}$ set point changes for dry, rainy and stormy weathers.

As it can be seen, the tracking of the $S_{O,5}$ reference signal is accurate for all three weather conditions since the controlled variable mostly overlaps the set point.

With these results, the first objective of this project is fulfilled: an IMC controller based on LSTM structures has been designed enhancing the default control of the dissolved oxygen of the fifth reactor tank.

4.2 ANN-IMC Control of the Nitrate and Nitrite Nitrogen of the Second Reactor Tank ($S_{NO,2}$)

In this section, another IMC controller based on ANNs is designed. It is devoted to controlling the nitrate and nitrite nitrogen of the second reactor tank ($S_{NO,2}$) of the BSM1 with higher performance than the original PI controller dedicated to this matter. Consequently, the goal of this module is to complete the schematic shown in Figure 4.9.

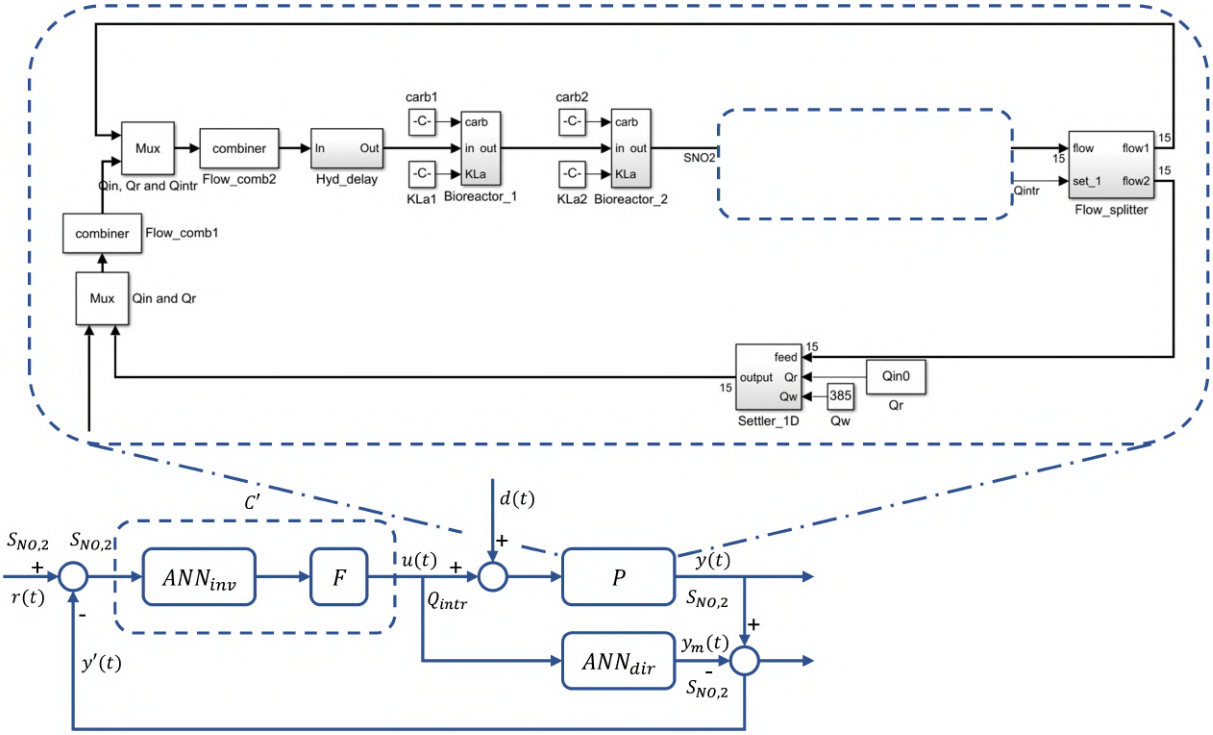


Figure 4.9: ANN-IMC controller structure for $S_{NO,2}$.

The big difference between this section and the previous one, apart from the controlled variable, is the real process to be controlled. As it is indicated, the real process P of this schematic is considerably more complex than the one of the dissolved oxygen. Now P not only corresponds to a single bioreactor tank but to two tanks along with other subsystems such as a flow splitter or a settler. For the filter F, the same coefficients as the default $S_{NO,2}$ PI controller will be used considering that this predefined structure also incorporates a filter, so that the comparison between original and proposed controllers is fair and it is not F what makes the difference in the results but the models that reproduce the direct and inverse processes. The controlled variable, associated with $r(t)$, will be now $S_{NO,2}$, while the actuation variable, corresponding to $u(t)$, will become Q_{intr}, linked so that the controllers track the $S_{NO,2}$ variations modifying the Q_{intr} rate. Like in the previous module, the effort of this section is based on finding the ANN_{dir} and the ANN_{inv} that obtain a model output signal $y_m(t)$ with the greatest similarity

to the process output signal $y(t)$, thus minimizing the effects of process-model mismatch.

4.2.1 Direct and Inverse ANNs

For this new challenge, the two years of data are obtained from the BSM1 framework simulated in an open loop configuration when inputting the two years of influent alternating dry, rainy, and stormy weather together with the filtered version of a two-years Q_{intr} variable. Q_{intr} consists in normally distributed random numbers between 0 and 200000 with variations every 2 days given the nature of this signal [Ale08].

Due to the complexity of the process monitored by the second control loop, more information (a larger number of variables) will be needed to model it, so the study of correlation and mutual information (MI) is not only done from concentrations gathered at a single point in the plant as for the first control loop, but from four different points: the input and output of the first reactor tank, the output of the first combiner, and the output of the settler, all of them computed together with Q_{intr} and $S_{NO,2}$. This results in a total of four correlation matrices plus four MI matrices, and many possible combinations of features to form the ANNs. After several experiments with different data selections (the iterative process of development and refinement, presented in Figure 3.1), a total amount of 11 parameters were chosen with the help of data analysis: $S_{S,0}$ and $S_{ND,0}$ from the input of the first tank, $X_{S,1}$, $S_{O,1}$, $S_{NO,1}$, $S_{NH,1}$ and $S_{ALK,1}$ from the output of the first tank, $S_{NO,comb1}$ and $X_{ND,comb1}$ from the output of the first combiner, and $X_{B,H,settler}$ and $TSS_{settler}$ from the output of the settler. These are the shared concentrations to train the ANN_{dir} and the ANN_{inv}. Then to model the direct process in question, Q_{intr} is needed because it is the control handle to manage the $S_{NO,2}$, and Q_{in} is also used to be included in the PI architecture, and so to keep similarity. To shape the inverse process, besides considering the 11 variables listed above, $S_{NO,2}$ becomes another input, Q_{in} is also taken into account, and finally a delayed version of the manipulated variable Q_{intr} is adopted, as it was done in the previous section to facilitate the modeling of the inverse process.

Once selected the information that will be used to train the ANNs, its preprocessing is done applying a sliding window with a window width of 10 time steps for the two ANNs to have the same initialization time of 2.5 hours of the first controller. Figure 4.10 shows the outcome of implementing this technique. The values of $S_{NO,2}$ and Q_{intr} from the tenth time step forward, will be assigned to each input example of the ANN_{dir} and the ANN_{inv} respectively, so that the algorithms have input/output pairs from which to learn.

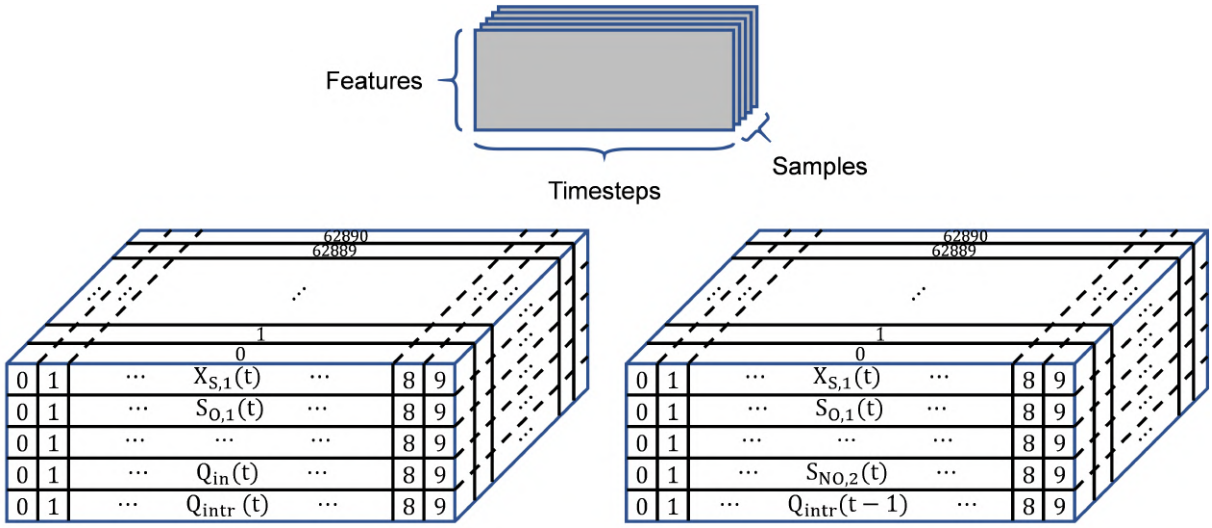


Figure 4.10: Shape of the 3D input arrays used to train the LSTM networks of the direct (left) and inverse (right) processes of the second control loop.

With the samples ready to train the networks, it is time to define their architecture. LSTM cells will be used since they are one of the most suitable algorithms for time series problems. A grid search of the parameters and layouts with which to build models that provide accurate predictions is performed, Table 4.4 summarizes the structure of the LSTM networks chosen to form the second controller.

LSTM structures				
Model layer	**Type**	**Neurons per gate**	**Activation function**	**Regularizer**
1st	LSTM	50	*Tanh & sigmoid*	L2 penalty (0.001)
2nd	Dense	1	*Linear*	-

Table 4.4: LSTM structures for the ANN_{dir} and the ANN_{inv} of the $S_{NO,2}$ ANN-IMC controller.

Both the network of the direct and the inverse processes are formed with a single LSTM cell with 50 neurons per gate in the hidden layer, followed by one neuron with a linear activation function in the output layer. The only distinction between the two structures is in the input layer. It just takes the input signals (values) and passes them to the next layer for further processing by subsequent layers. So, since the number and type of signals used for the ANNs training are different for both architectures, these input layers will be too regarding the number of neurons. In short, in terms of network design, among all the possible combinations to form the four ANNs that make up the two IMCs, the same structure has been maintained for the two pairs of LSTM models that compose them. Only the data and the number of neurons per gate have varied. Once the remaining blocks representing the model of the real process and its inverse have been defined, the design of the $S_{NO,2}$ ANN-IMC controller can be completed as depicted in the sketch of Figure 4.11.

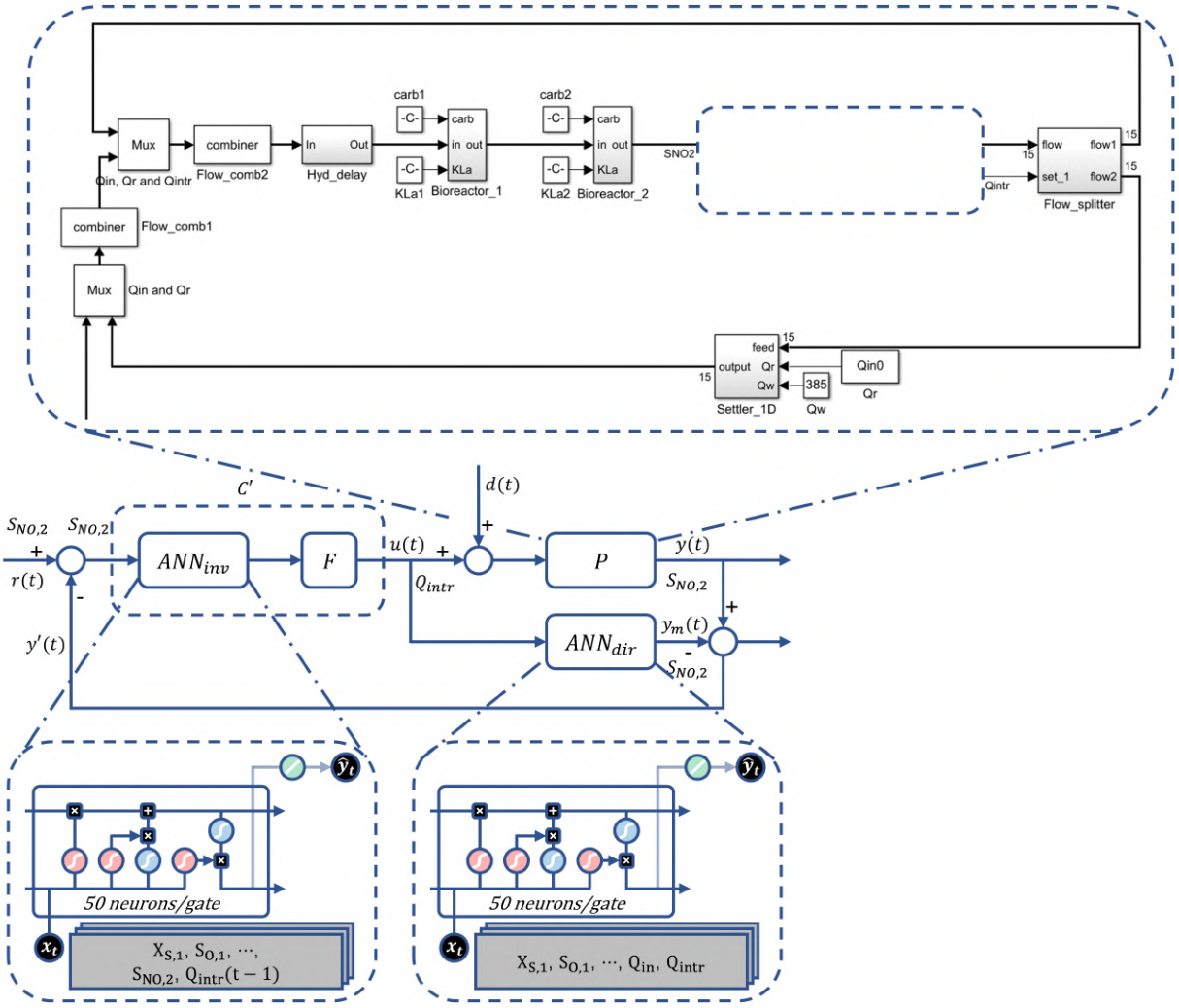


Figure 4.11: Proposed ANN-IMC controller structure for $S_{NO,2}$.

As for the $S_{O,5}$ controller, the L2 regularization technique is defined with a penalty of 0.001 to prevent model overfitting. In terms of model compilation, the Adam optimization algorithm is used again to train the network, together with the MSE loss function to evaluate it.

Once the networks are compiled, they are fit during the model training process, which is configured in the same way as the previous controller: 200 epochs, a batch size of 5000 samples, and a K-fold cross-validation of K = 9 (see Figure 3.9) in order to get nine models created with different sections of the whole dataset. This partition results in 80% of the data (55903 samples) for training, and 10% of the data (6988 samples) for both validation and test. These last parameters concerning the model compilation and model training are also unchanged with respect to the previous ANN-IMC. In this way, the necessary networks to design the new $S_{NO,2}$ controller can be built, and once created, it is possible to evaluate them with the five metrics defined in Section 3.3.

ANNs Performance

After training the required LSTM networks to model the process that goes from the entrance of the flow splitter to the exit of the second tank and its inverse, they are tested with data they have not seen before and evaluated to know how they work. In order to evaluate the functioning of the networks that will form the second controller, the MAPE, the MAAPE, the RMSE, the NRMSE, and the R^2 will be used again. However, only the MAAPE, the NRMSE, and the R^2 will serve to compare the new results with those of the previous section because they are scale independent and free of outliers for close-to-zero values. The MAPE and the RMSE will also be useful for assessing the performance of the latest models considering their individual range of values. It is worth to notice that it is very important to use multiple evaluation metrics to evaluate the models because they may perform well according to the measurement of one metric, but poorly when measured by another one [Eva]. The average of the scores of the nine experiments carried out by the 9-fold cross-validation (training and validation), together with the scores of the nine models impartially evaluated on the last 10% of the whole dataset (test), give rise to the numbers of Table 4.5.

Metric	**LSTM direct model**			**LSTM inverse model**		
	Training	**Validation**	**Test**	**Training**	**Validation**	**Test**
MAPE [%]	11.589	11.845	5.267	1.643	1.447	1.141
MAAPE [%]	4.912	4.930	3.416	3.670	3.671	5.401
RMSE	0.057	0.057	0.059	465.259	467.882	518.255
NRMSE	0.004	0.004	0.004	0.003	0.003	0.003
R^2	1.000	1.000	1.000	1.000	1.000	1.000

Table 4.5: Performance of the ANN_{dir} and the ANN_{inv} of the $S_{NO,2}$ ANN-IMC controller.

The values obtained from the five metrics for the nine models on the three dataset partitions were quite similar between folds. This indicates once again that there is a variety of examples throughout the dataset from which the nine models can be trained offering a similar operation for each fold due to their balanced plurality of examples.

From the average it can be observed a remarkable improvement of these results compared to those of the previous controller. R^2 is always one, which means that there is perfect correlation between real output and predicted values. This is also an indicator of a highly reliable model for future forecasts. NRMSE is almost zero and MAAPE is very low in both models, which is synonymous of almost perfect predictions. RMSE and MAPE corroborate the success of the aforementioned metrics showing also small values bearing in mind that the direct model predicts numbers of $S_{NO,2}$ between 0 and 16 g/m^3 while the inverse model predicts values of Q_{intr} between 0 and 200000 m^3. Here it can be appreciated the importance of MAAPE and NRMSE in supporting these metrics: the NRMSE serves to prove that an RMSE error of the

inverse model around 500 is low when working with a scale of values of the order of magnitude five, whereas the MAAPE serves to avoid imprecise measurements such as those of the MAPE of the direct model when working with values close to zero. Some MAPE and MAAPE values give unusual values for testing. But as explained in the previous section, this is due to the constraint of these metrics for small actual values, and to the shape of the signal in that period. Comparing the four ANNs designed to form the IMCs, it has been confirmed that the models composing the $S_{NO,2}$ ANN-IMC controller work with greater precision than those composing the $S_{O,5}$ ANN-IMC controller. Among the models forming the $S_{NO,2}$ ANN-IMC controller, the MAAPE, NRMSE, and R^2 reveals that the inverse models generally perform a little bit better that the direct ones due to the extra feature of the delayed output.

Given that the nine models generated during training for each process showed similar measurements in their evaluation, any combination of folds of the direct and inverse models could be appropriate to form the controller. In particular, the 1st fold was chosen for both the ANN_{dir} and the ANN_{inv} for generalizing and performing satisfactorily once incorporated into the $S_{O,5}$ ANN-IMC controller structure. Figure 4.12 shows a visual example of how well the ANNs work.

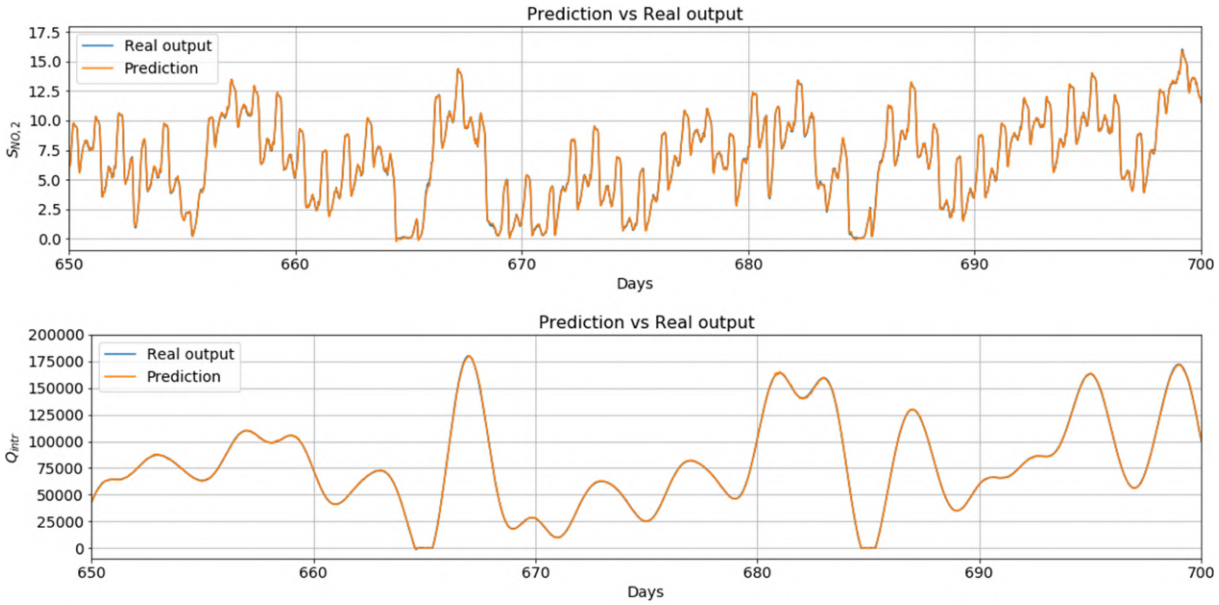


Figure 4.12: Predictions from day 650 to day 700 of the ANN_{dir} (top), and the ANN_{inv} (bottom) of the $S_{NO,2}$ ANN-IMC controller.

The above fifty-day plots of the $S_{NO,2}$, and the Q_{intr} demonstrates that the predictions are so precise that the blue trace representing the actual values is barely visible because it is overlapped by the orange trace representing the forecasts. Knowing that the models of the real process and its inverse are performing excellently, it is time to export their weights and biases along with the means and variances from Python, to import them in MATLAB, where the $S_{NO,2}$ ANN-IMC controller will be designed and tested once implemented in the BSM1.

4.2.2 ANN-IMC Controller

Once the model parameters are extracted in Python, they can be loaded on other frameworks such as MATLAB and Simulink. In the same way as for the previous controller, first the networks of the direct and inverse processes are implemented separately, as a strategy to ensure the same excellent operation that was obtained in the previous subsection by comparing the forecasts of each LSTM network in both platforms. Otherwise, if the controller is directly created in Simulink, there are so many elements that compose the ANN-IMC structures that searching for an error could become a tedious task. Figure 4.13 shows the system that emulates the behavior of the ANN_{dir} in Simulink. As in the previous section, only the layout of the network structure of the direct process will be exposed, considering that the one of the inverse process will be practically identical unlike the input data.

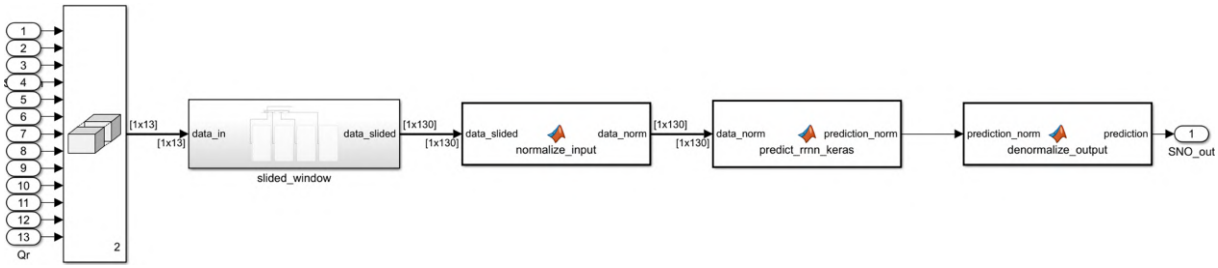


Figure 4.13: Interface layout of the ANN_{dir} structure of the $S_{NO,2}$ ANN-IMC controller in Simulink.

As it is noted, the elements that make up the ANNs of this second controller in Simulink are practically the same as those observed in the controller of the previous section. The major difference between them is found in the initial block and its number of input signals. Since the number of variables with which the ANN_{dir} and ANN_{inv} of the $S_{NO,2}$ were trained is 13 and 14 for each network, the same concentrations are used now as inputs. The horizontal concatenation of these entries results in 1x13 and 1x14 signals for the direct and inverse models, respectively.

The following blocks follow the same procedure seen for the first controller. After grouping the incoming data, a first preprocessing step is applied, which consists of the sliding window described by the schematic of Figure 4.14.

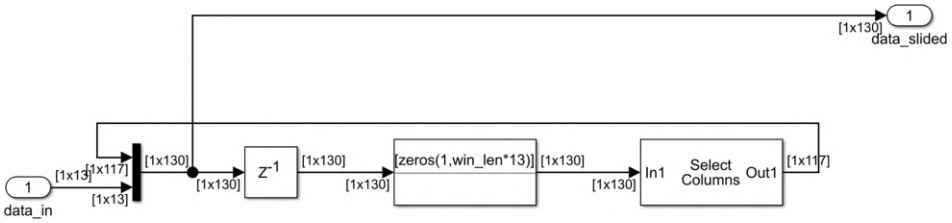


Figure 4.14: Interface layout of the sliding window of the $S_{NO,2}$ ANN_{dir} structure in Simulink.

The previous subsystem basically orders the data in such a way that the network itself (implemented in a subsequent block) receives the data of each variable for 10 time step (the window width defined). Its operation follows the same steps as those in the sliding window of the previous section. The difference is in the data considered as well as their mean and variance.

Following the structure, data is scaled before and after the core of the ANN structures. The MATLAB code that defines the operations of LSTM networks is defined within the box after preprocessing. Besides, here the function is programmed for calculations with 50 neurons per gate in order to map the received inputs to output values with the same accuracy as was shown in the previous subsection.

Having the subsystems of the $S_{NO,2}$ ANN_{dir} and ANN_{inv} structures, the second controller can be assembled as seen in Figure 4.15.

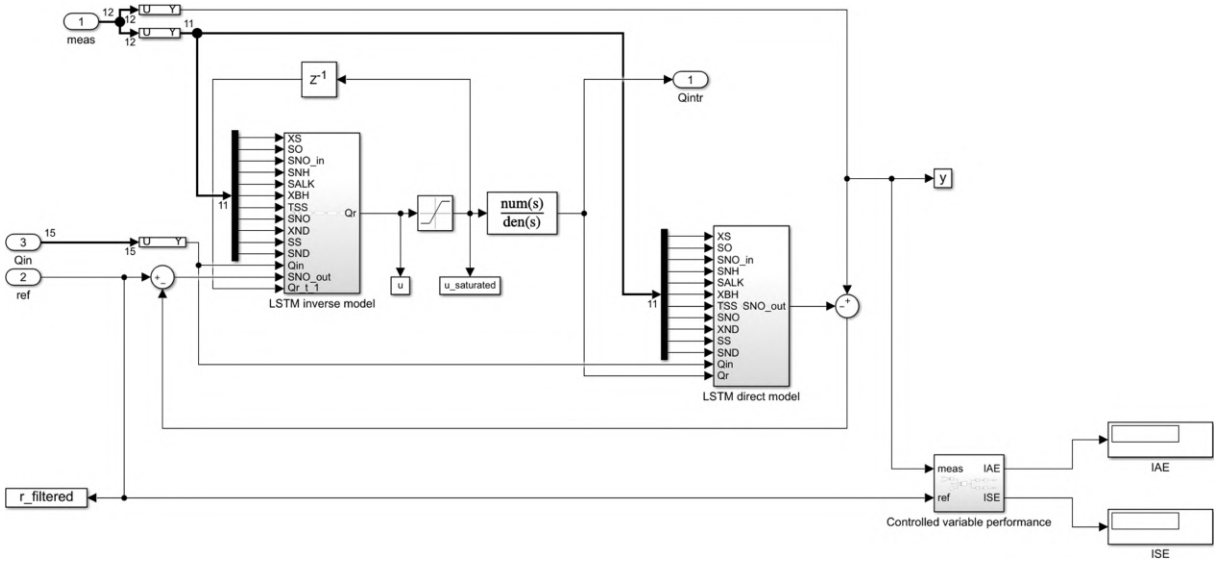


Figure 4.15: Interface layout of the $S_{NO,2}$ ANN-IMC controller in Simulink.

As it is observed, the controller layout follows the same structure presented in Figure 4.11, and visually differs from the previous controller in the number and type of input signals of each LSTM network. However, this time the first-order filter located in the middle system is a transfer function with gain and constant time set to 1 and 1/1000 respectively, the same numbers taken from the filter of the Q_{intr} PI controller. The same criteria is applied to the saturation block after the ANN_{inv}, which just limits the predictions of Q_{intr} in the same way as the default PI does. Input and output ports 1 refer to the input and output of the process that goes from the entrance of the flow splitter to the exit of the second tank. Finally, the subsystem at the bottom right performs the calculation of the IAE and ISE metrics to determine the performance of the proposed controller.

Controller Performance

The default PI controller as well as the ANN-IMC of the $S_{NO,2}$ will be evaluated to compare the performance of the default control strategy with the new proposed IMC one. The same reference signal is employed for this assessment, so that the measurements and the controllers are evaluated from the same point of view. The $S_{NO,2}$ concentration generated for this purpose is a random square pulse signal of 12-hour variations from 0 to 14 g/m^3, which describes a shape that LSTM networks know how to work with. Again, the performance is computed in terms of the IAE and ISE over the last 7 days of tracking. For that purpose, BSM1 influent profiles with dry, rain and storm weathers are considered. Table 4.6 provides the operation results of the default PI controller and the ANN-IMC controller of the second control loop for a $S_{NO,2}$ signal with the features mentioned above.

Metric	**PI**			**IMC**			**Improvement [%]**		
	Dry	**Rain**	**Storm**	**Dry**	**Rain**	**Storm**	**Dry**	**Rain**	**Storm**
IAE	5.567	9.663	8.211	4.546	8.736	7.253	18.34	9.59	11.67
ISE	11.580	24.180	20.320	5.867	17.830	15.020	49.34	26.26	26.08

Table 4.6: Performance of the PI controller and the ANN-IMC controller of the $S_{NO,2}$.

The percentage of improvement is computed to exhibit the reduction in the IAE and ISE of the $S_{NO,2}$ ANN-IMC controller with respect to the original PI implemented in the BSM1. These results show that the $S_{NO,2}$ ANN-IMC controller offers the best performance for all three climates. A decrease of 1.021 in IAE and 5.713 in ISE for dry weather represents the best scenario with an improvement of 18.34% and 49.34% accordingly. For rainy and stormy weathers, the gains are not as high as in the previous climate. Nevertheless, an enhancement in IAE and ISE of 9.59% and 26.26% has been achieved for rain weather, while for storm the upgrade has been 11.67% and 26.08%. In average, the scores account for 13.20% in IAE, and 33.89% in ISE, which prove that the nitrate and nitrite nitrogen control strategy by default is notably improved.

Despite achieving a better performance in the $S_{NO,2}$ ANNs than in the $S_{O,5}$ ANNs, the improvement in the performance of the $S_{O,5}$ controller is larger than the one of the $S_{NO,2}$ controller. This situation is due to PI controllers, which are designed adopting a linearization of the real process [Pis19a]. Therefore, a fixed or slowly varying set point (as is the case of the $S_{NO,2}$ reference signal) will result in a more linear process to be monitored, and consequently preferable for this type of controllers. Considering that the set point and the process under control of each problem are completely different (see Figures 4.3 and 4.11), the baseline scenario is distinct too, so the improvements of the proposed structures are not comparable.

Finally, Figure 4.16 plots the tracking of the $S_{NO,2}$ set point for the different weathers by the proposed IMC controller.

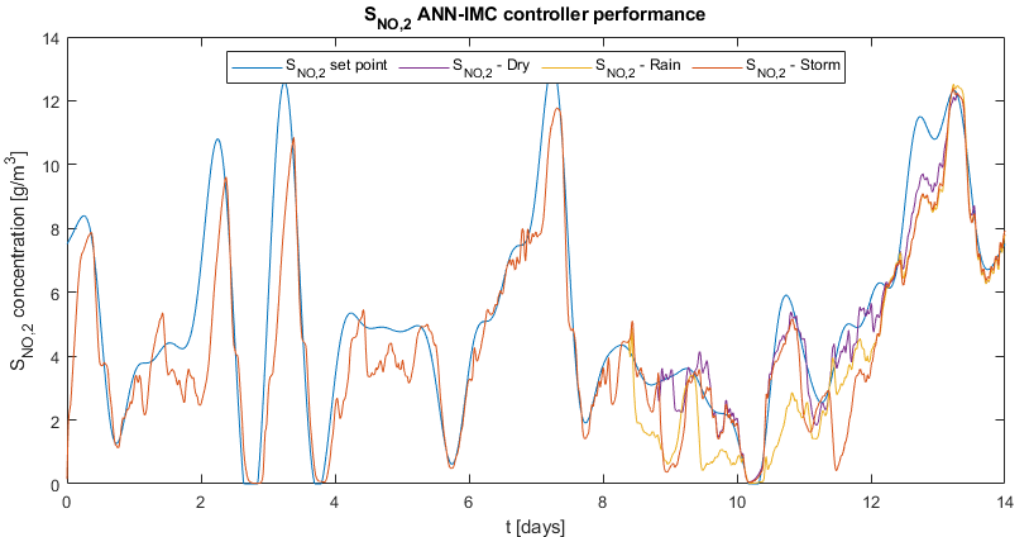


Figure 4.16: Tracking of $S_{NO,2}$ set point changes for dry, rainy and stormy weathers.

As it is observed, the $S_{NO,2}$ at the process output follows the changes of the reference signal for all the three weather conditions, despite the difficulties presented by rain and storm conditions around days 8 to 12.

It is worth to notice that with the ANN-based IMC strategy implemented, a very precise representation of the process being controlled and its inverse (see Figure 4.12), is not sufficient to achieve perfect control as indicated by the internal model principle. This does not mean that the theory is wrong, but that there is an additional factor such as the feedback error (the difference between the process and model outputs), which is not modeled by the ANNs (it is not considered in the training dataset) and consequently affects the functioning of the strategy presented in this dissertation. To alleviate this, the addition of certain level of noise to the data (especially to the reference signal) can be considered to include this factor (the sum or difference operation at the beginning of the ANN-IMC controllers) in the model training.

With these results, the second objective of this project is fulfilled: an IMC controller based on LSTM structures has been designed enhancing the default control of the nitrate and nitrite nitrogen of the second reactor tank. The product of this section, combined with the previous one, opens the door to the last part of development and results analysis.

4.3 ANN-IMC Control of the Overall Benchmark Simulation Model no. 1 (BSM1)

This final part of Chapter 4 will merge the two ANN-IMC controllers designed separately in the two preceding sections to replace the default structures of the BSM1 as indicated in Figure 1.2. In this way, $S_{O,5}$ and $S_{NO,2}$ concentrations will be managed exclusively with LSTM-type neural

networks, and thus solely with data.

The original Simulink plant layout that was illustrated in Figure 2.2, looks like the following system once the modifications are applied.

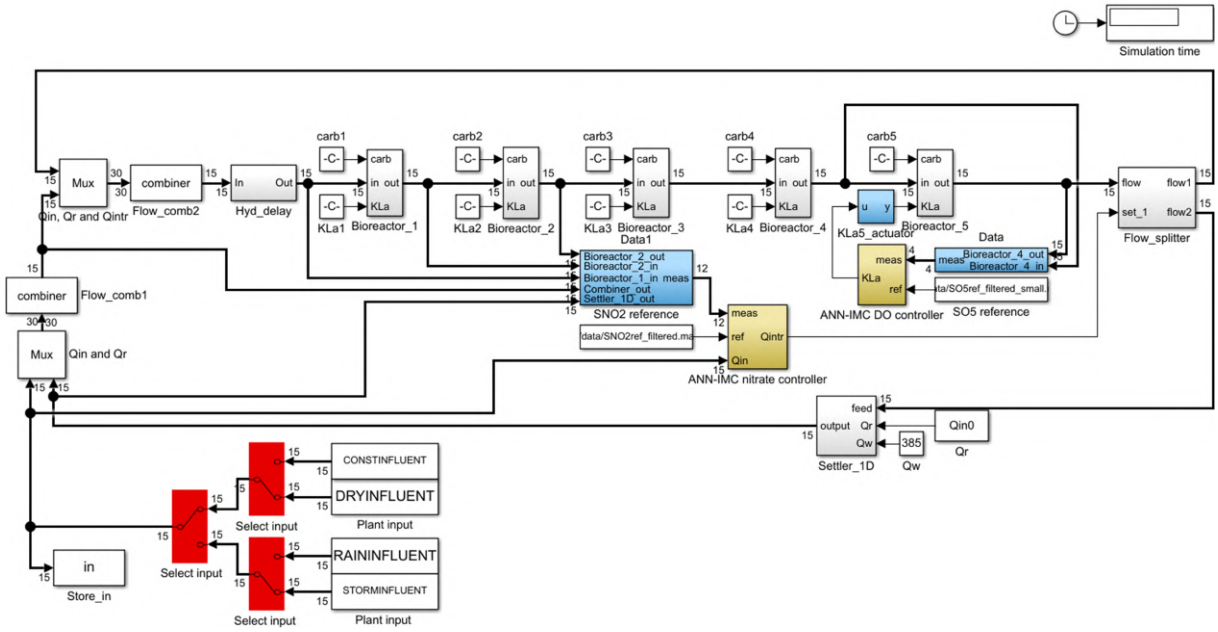


Figure 4.17: Interface layout of the new BSM1 plant in Simulink.

The subsystems that define the ANN-IMC controllers for dissolved oxygen, and nitrate and nitrite nitrogen, contain the structures that were shown in Figure 4.7 and Figure 4.15 respectively. The points of the plant where input information is taken coincide with those carrying signals that were used to train the ANNs of both controllers. For instance, in the case of the $S_{NO,2}$, 13 variables for the ANN_{dir} and 14 variables for the ANN_{inv} were chosen to model the real process, encompassing the flow splitter along with the first two tanks, and its inverse. Various points across the plant were considered to take these concentrations and model the processes more precisely: the input and output of the first reactor tank, the output of the first combiner, the output of the settler, and the input of the plant (to include Q_{in}). Also, the output of the second reactor tank is required to include the $S_{NO,2}$ as the output of the process under control, and estimate the effect of disturbances and model mismatch.

Once the ANN-IMC controllers have been implemented in the BSM1 plant, their behavior can be evaluated to prove the plant enhancement and see the controllers dependence.

Controllers Performance

Next, the operation of both default PI controllers and of both proposed IMCs will be evaluated in order to contrast the functioning of both strategies in the monitoring of an entire WWTP scenario.

An important factor in joining the two controllers developed separately in the previous sections is the impact of one structure on the other. Initially, each of these ANN-IMCs was designed maintaining the original PI of the other concentration. The default set points of the $S_{O,5}$ and $S_{NO,2}$ in BSM1 framework are fixed at 2 g/m^3 and 1 g/m^3 respectively. It means that the $S_{O,5}$ ANNs were trained with data collected from the plant when the $S_{NO,2}$ control loop had a fixed set point of 1 g/m^3. In the same way that the $S_{NO,2}$ ANNs were trained with data collected from the plant when the $S_{O,5}$ control loop had a fixed set point of 2 g/m^3. So, the networks that conform the controllers have been trained under these configurations, in which the other structure employs a constant reference signal, and thus, joining the proposed ANN-IMCs with variable set points will affect their operation. To mitigate this issue without having to rebuild the networks for a plant with variable set points, a balance has been sought between reference signals: the same set point of Section 4.2 is used for the $S_{NO,2}$, while a random square pulse signal of 1-hour variations from 1.5 to 2.5 g/m^3 of $S_{O,5}$ (instead of 0 to 5 g/m^3 as in Section 4.1) is considered. In that manner, the $S_{O,5}$ set point is closer to the default value of 2 g/m^3 without getting too far from the variations with which the $S_{O,5}$ ANNs were trained. As in the preceding sections, the performance is computed for each controller considering the three BSM1 influent profiles and the IAE and ISE metrics over the last 7 days of control. Table 4.7 provides the scores of default PI controllers and ANN-IMC controllers of first and second control loops.

$S_{O,5}$									
Metric	**PI**			**IMC**			**Improvement [%]**		
	Dry	**Rain**	**Storm**	**Dry**	**Rain**	**Storm**	**Dry**	**Rain**	**Storm**
IAE	0.337	0.306	0.332	0.335	0.309	0.317	0.50	0.00	4.40
ISE	0.038	0.034	0.037	0.037	0.031	0.032	2.08	7.83	11.58
$S_{NO,2}$									
Metric	**PI**			**IMC**			**Improvement [%]**		
	Dry	**Rain**	**Storm**	**Dry**	**Rain**	**Storm**	**Dry**	**Rain**	**Storm**
IAE	5.563	9.668	8.199	5.370	9.577	8.384	3.47	0.94	0.00
ISE	11.620	24.160	20.340	7.542	19.780	17.380	35.09	18.13	14.55

Table 4.7: Performance of the PI controllers and the ANN-IMC controllers of the $S_{O,5}$ and the $S_{NO,2}$.

Focusing directly on the percentage of improvement, it can be appreciated the importance of using networks under the same circumstances (same kind of data) they were trained. In this third stage with the controllers combined, the upgrade is not as great as the one achieved for the controllers separately. Nevertheless, the strategy approach of this thesis still improves the preestablished one in most measurements. The results show that both ANN-IMC controllers offers best performance for all three climates in terms of ISE: 7.16% of average improvement

in tracking the $S_{O,5}$, and 22.59% for the $S_{NO,2}$. Regarding the IAE, the scores show a modest enhancement in two of the three weathers for each structure: 0.50% and 4.40% of reduction in dry and stormy conditions for the $S_{O,5}$, while 3.47% and 0.94% of reduction in dry and rainy periods for the $S_{NO,2}$. The remaining climates show similar results to those of the PI controllers in terms of the IAE.

As a summary, the results are also satisfactory even though the improvements are not as significant as the ones achieved individually. The implemented control approach can be regarded as an alternative to the one fixed by default, also providing more precise tracking. For specific weathers where there is no improvement in IAE, there is a trade-off between metrics: a reduction in ISE results in a control system that produces fewer large errors. In the context of WWTPs, it is vitally important to avoid large big mistakes that involve exceeding the established limits for some concentrations, being preferable to commit certain small errors. Finally, and most important, these measurements can be enhanced by training the networks for this scenario as discussed before (rebuilding the networks for a plant with variable set points). However, this work is beyond the scope of this thesis.

With these results, the third objective of this project is fulfilled: a control system using IMC controllers based on LSTM structures has been established enhancing the default control strategy.

Chapter Summary

As it is demonstrated throughout this chapter, all the objectives listed in the introduction have been fulfilled. Two of the most used metrics in the field of control (IAE and ISE) have been employed to prove the improved performance of the proposed controllers over the default ones. Between this chapter and the previous one, it has been possible to reflect the magnitude of the challenge involved in programming ANNs: there are an infinite number of configurations to create the models on another large number of available data. Every decision taken in the formation of these networks and therefore in the controllers, matters a lot in the final results. Finally, the separately created controllers have been combined into a single benchmark scenario to have a WWTP based exclusively on ANNs and data. It has been seen that despite the degradation by the effect of one controller on the other, the new strategy generally improves the original one. Besides, it has been indicated how these results could be enhanced following the same guidelines of this project.

Chapter 5

Conclusions and Future Work

This dissertation has implemented an ANN-based IMC strategy in BSM1 WWTP simulation scenario, where it has been possible to analyze the proposed control approach. It was based on ANNs (LSTM structures) to model the processes required by a standard IMC (the process under control and its inverse). Thus, the commitment of this work mainly turned around the design of ANNs, which are the main blocks that constitute the IMC controller of the proposed concept. The workflow for the creation of ANNs models was detailed in Chapter 3.

Two ANN-IMC controllers have been developed in Chapter 4 as an alternative to the default PI controllers of the BSM1. The behavior of the strategy adopted in this project and the one established by default in the BSM1 has been compared by means of the IAE and ISE metrics considering influent profiles of dry, rain, and storm weathers.

- In Section 4.1 a first controller was designed to track the dissolved oxygen in the fifth reactor tank ($S_{O,5}$) concentration at a variable set point actuating over the oxygen transfer coefficient (K_La). Results show that the $S_{O,5}$ ANN-IMC controller improves the PI behavior for the three analyzed climates in an average 43.90% of IAE, and 82.64% of ISE.

- In Section 4.2 a second controller was designed to track the nitrate and nitrite nitrogen in the second reactor tank ($S_{NO,2}$) concentration at a variable set point actuating over the internal recirculation flow rate (Q_{intr}). Results show that the $S_{NO,2}$ ANN-IMC controller improves the PI behavior for the three analyzed climates in an average 13.20% of IAE, and 33.89% of ISE.

Moreover, in Section 4.3 both controllers were joined together on the same BSM1 scenario to provide a control system exclusively managed by ANNs. The results of both ANN-IMC controllers generally improved the PIs behavior for the chosen set points in IAE and ISE. However, their performance has been influenced by working in a scenario where the information processed

by the ANNs differs from that used in their training (ANN models are built to deal with a particular problem).

In general, the development of this thesis has contributed in some way to the study of new measures to mitigate the problems of environmental pollution by improving the operation of WWTPs, in addition to favoring the expenses of these plants. In turn, this has demonstrated that ANNs are capable of modeling complex and non-linear processes with great precision and only with data.

Although the contribution of this dissertation ends here, new fronts always appear. A list of some topics not covered in this thesis that may be of interest for further studies is given below.

- **Improve the accuracy of the ANNs, and hence the operation of the controllers**. As discussed throughout this project, the control quality of an IMC structure is directly related to the model of the process being controlled and its inverse. For the proposed control approach, ANNs are adopted to model both processes. However, there are many aspects to consider in their construction, such as the architecture of the model itself or the data to be used. Although precise prediction and tracking results have been achieved in this dissertation, there are uncountable number of combinations to form the ANNs, and consequently proposals for new ANN-IMC controllers improving the current results.

- **Improve the operation of the controllers once they are joined**. As mentioned in Section 4.3, the union of the separately created controllers with different set points affects the operation of both ANN-IMCs. Despite this, the scores obtained were quite good, and most importantly they can be enhanced by training the networks for a scenario where the properties of the other controller are considered.

- **Test the behavior of the controllers when real sensors are considered**. Data gathering and tracking of concentrations have been carried out when ideal sensors (without noise) are adopted for both PI and ANN-IMC controllers. Nevertheless, there is the possibility of working in a more realistic scenario using real sensors. This line of study will also allow to analyze which strategy is more robust against noise.

- **Implement the ANN-IMC strategy on the BSM2 or in other industries**. BSM2 framework includes the BSM1 for the biological treatment of the wastewater in addition to the sludge treatment. This means working with a more complete scenario, with more sophisticated processes, and more like real WWTPs. It is important not to limit the ANN-IMC control approach to the specific context of this work, but it is also interesting to implement this strategy in BSM2, as well as in other industries that require the control of a system.

- **Deploy the controllers in a real WWTP**. There is not a better way to ensure that the ANN-IMC strategy works properly than implementing it in a real WWTP. This project has carried out a systematic process and created reliable and accurate models that can make predictions for IMC structures. The next step would be to put these models forming the ANN-IMC controllers in operational software to bring the contributions of this dissertation to the real world.

Bibliography

[Ale99] Jens Alex, JF Beteau, JB Copp, C Hellinga, Ulf Jeppsson, S Marsili-Libelli, MN Pons, H Spanjers, H Vanhooren, "Benchmark for evaluating control strategies in wastewater treatment plants", *1999 European Control Conference (ecc)*, pags. 3746–3751, IEEE, 1999.

[Ale08] J Alex, L Benedetti, J Copp, KV Gernaey, Ulf Jeppsson, I Nopens, MN Pons, L Rieger, C Rosen, JP Steyer, *et al.*, "Benchmark simulation model no. 1 (bsm1)", *Report by the IWA Taskgroup on benchmarking of control strategies for WWTPs*, pags. 19–20, 2008.

[Bar18] Marian Barbu, Ignacio Santin, Ramon Vilanova, "Applying control actions for water line and sludge line to increase wastewater treatment plant performance", *Industrial & Engineering Chemistry Research*, Vol. 57, nº 16, pags. 5630–5638, 2018.

[Ber18] Christoph Bergmeir, Rob J Hyndman, Bonsoo Koo, "A note on the validity of cross-validation for evaluating autoregressive time series prediction", *Computational Statistics & Data Analysis*, Vol. 120, pags. 70–83, 2018.

[Bon17] Massimiliano Bonamente, *Statistics and Analysis of Scientific Data*, Springer, 2017.

[Bro19a] Jason Brownlee, "Difference between a batch and an epoch in a neural network", https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/, oct. 2019, accessed: 2020-05-28.

[Bro19b] Jason Brownlee, "Overfitting and underfitting with machine learning algorithms", https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/, aug. 2019, accessed: 2020-05-25.

[Bro19c] Jason Brownlee, "The promise of recurrent neural networks for time series forecasting", https://machinelearningmastery.com/promise-recurrent-neural-networks-time-series-forecasting/, aug. 2019, accessed: 2020-05-28.

[Bro19d] Jason Brownlee, "What is deep learning?", https://machinelearningmastery.com/what-is-deep-learning/, aug. 2019, accessed: 2020-05-25.

[Bro20] Jason Brownlee, "Difference between algorithm and model in machine learning", https://machinelearningmastery.com/difference-between-algorithm-and-model-in-machine-learning/, apr. 2020, accessed: 2020-05-25.

[Cho17] Francois Chollet, *Deep Learning with Python*, Manning, 2017.

[Cli] "Climate change", https://www.nationalgeographic.com/environment/climate-change/, accessed: 2020-05-15.

[Com91] European Commission, "Council directive 91/271/eec of 21 may 1991 concerning urban waste water treatment", *OJ*, pags. 40–52, 1991.

[Con13] Henry Concepción, Darko Vrecko, Montserrat Meneses, Ramón Vilanova, "Control strategies for removing nitrogen compounds in wastewater treatment plants", *2013 9th Asian Control Conference (ASCC)*, pags. 1–6, IEEE, 2013.

[Cop02] John B Copp, *The COST Simulation Benchmark: Description and Simulator Manual: a Product of COST Action 624 and COST Action 682*, EUR-OP, 2002.

[Cov12] Thomas M Cover, Joy A Thomas, *Elements of information theory*, John Wiley & Sons, 2012.

[Eva] "Evaluation metrics", https://deepai.org/machine-learning-glossary-and-terms/evaluation-metrics, accessed: 2020-06-02.

[Fre99] Yoav Freund, Robert E Schapire, "Large margin classification using the perceptron algorithm", *Machine learning*, Vol. 37, nº 3, pags. 277–296, 1999.

[Gan13] Srungarapu Naga Venkata Ganesh, "A unified and respective algorithm for injecting voltage in 3-phase series power quality controller", 2013.

[Gar82] Carlos E Garcia, Manfred Morari, "Internal model control. a unifying review and some new results", *Industrial & Engineering Chemistry Process Design and Development*, Vol. 21, nº 2, pags. 308–323, 1982.

[Ger14] Krist V Gernaey, Ulf Jeppsson, Peter A Vanrolleghem, John B Copp, *Benchmarking of control strategies for wastewater treatment plants*, IWA Publishing, 2014.

[Goo16] Ian Goodfellow, Yoshua Bengio, Aaron Courville, *Deep learning*, MIT press, 2016.

[Hen87] M Henze, CPL Grady, W Gujer, GvR Marais, T Matsuo, "Activated sludge model no. 1. iawprc task group on mathematical modelling for design and operation of biological wastewater treatment", *IAWPRC Scientific and Technical Reports, London*, 1987.

[Hoc97] Sepp Hochreiter, Jürgen Schmidhuber, "Long short-term memory", *Neural computation*, Vol. 9, nº 8, pags. 1735–1780, 1997.

[Jep07] Ulf Jeppsson, M-N Pons, Ingmar Nopens, J Alex, JB Copp, KV Gernaey, Christian Rosén, J-P Steyer, PA Vanrolleghem, "Benchmark simulation model no 2: general protocol and exploratory case studies", *Water Science and Technology*, Vol. 56, nº 8, pags. 67–78, 2007.

[Kan18] Nandha Kumar Kandasamy, Giridharan Karunagaran, Costas Spanos, King Jet Tseng, Boon-Hee Soong, "Smart lighting system using ann-imc for personalized lighting control and daylight harvesting", *Building and Environment*, Vol. 139, pags. 170–180, 2018.

[Ker] "Keras. simple. flexible. powerful.", https://keras.io/, accessed: 2020-05-10.

[Kim16] Sungil Kim, Heeyoung Kim, "A new metric of absolute percentage error for intermittent demand forecasts", *International Journal of Forecasting*, Vol. 32, nº 3, pags. 669–679, 2016.

[MAT] "What is matlab?", https://www.mathworks.com/discovery/what-is-matlab.html, accessed: 2020-05-10.

[Men16] Montse Meneses, Henry Concepción, Ramon Vilanova, "Joint environmental and economical analysis of wastewater treatment plants control strategies: a benchmark scenario analysis", *Sustainability*, Vol. 8, nº 4, pags. 360, 2016.

[Mit97] Tom Mitchell, *Machine Learning*, McGraw-Hill, 1997.

[Nai17] Aakash Nain, "Tensorflow or keras? which one should i learn?", https://medium.com/implodinggradients/tensorflow-or-keras-which-one-should-i-learn-5dd7fa3f9ca0, may 2017, accessed: 2020-05-10.

[Ng15] Andrew Ng, "What data scientists should know about deep learning", https://www.slideshare.net/ExtractConf/andrew-ng-chief-scientist-at-baidu, nov. 2015, accessed: 2020-05-18.

[Nun20] Christina Nunez, "Water pollution is a rising global crisis. here's what you need to know.", https://www.nationalgeographic.com/environment/freshwater/pollution/, jan. 2020, accessed: 2020-05-15.

[O'L13] D. E. O'Leary, "Artificial intelligence and big data", *IEEE Intelligent Systems*, Vol. 28, nº 2, pags. 96–99, 2013.

[Pis19a] Ivan Pisa, Antoni Morell, Jose Lopez Vicario, Ramon Vilanova, "Ann-based internal model control strategy applied in the wwtp industry", *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pags. 1477–1480, IEEE, 2019.

[Pis19b] Ivan Pisa, Ignacio Santín, Antoni Morell, Jose Lopez Vicario, Ramon Vilanova, "Lstm-based wastewater treatment plants operation strategies for effluent quality improvement", *IEEE Access*, Vol. 7, pags. 159773–159786, 2019.

[Pis19c] Ivan Pisa, Ignacio Santín, Jose Lopez Vicario, Antoni Morell, Ramon Vilanova, "Ann-based soft sensor to predict effluent violations in wastewater treatment plants", *Sensors*, Vol. 19, nº 6, pags. 1280, 2019.

[Pyt] "Python description", https://anaconda.org/anaconda/python, accessed: 2020-05-10.

[RNN20] "tf.compat.v1.nn.rnn_cell.rnncell", https://www.tensorflow.org/api_docs/python/tf/compat/v1/nn/rnn_cell/RNNCell, jun. 2020, accessed: 2020-05-28.

[Roj16] Jose David Rojas, Orlando Arrieta, Montse Meneses, Ramon Vilanova, "Data-driven control of the activated sludge process: Imc plus feedforward approach", *International Journal of Computers Communications & Control*, Vol. 11, nº 4, pags. 522–537, 2016.

[Rus10] Stuart Russell, *Artificial intelligence : a modern approach*, Prentice Hall, 2010.

[She08] Wenhao Shen, Xiaoquan Chen, Jean Pierre Corriou, "Application of model predictive control to the bsm1 benchmark of wastewater treatment process", *Computers & Chemical Engineering*, Vol. 32, nº 12, pags. 2849–2856, 2008.

[Sie97] Hava T Siegelmann, Bill G Horne, C Lee Giles, "Computational capabilities of recurrent narx neural networks", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, Vol. 27, nº 2, pags. 208–215, 1997.

[Sut18] Richard S Sutton, Andrew G Barto, *Reinforcement learning: An introduction*, MIT press, 2018.

[Tor18] Jordi Torres, *DEEP LEARNING Introducción práctica con Keras*, Lulu.com, 2018.

[Wan18] Chunyan Wang, Wanzhong Zhao, Zhongkai Luan, Qi Gao, Ke Deng, "Decoupling control of vehicle chassis system based on neural network inverse system", *Mechanical Systems and Signal Processing*, Vol. 106, pags. 176–197, 2018.

[Zel94] Andreas Zell, *Simulation neuronaler Netze*, Addison-Wesley, 1994.

[Zha20] Aston Zhang, Zachary C. Lipton, Mu Li, Alexander J. Smola, *Dive into Deep Learning*, 2020, `https://d2l.ai`.

Appendix A

ANNs Performance

A.1 Learning Curves

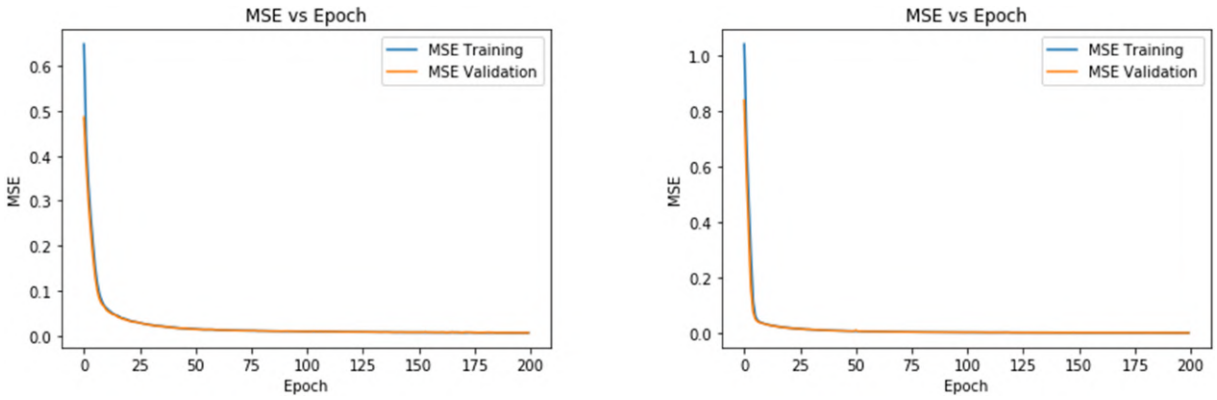


Figure A.1: Learning curves of the ANN_{dir} (left) and the ANN_{inv} (right) of the $S_{O,5}$ ANN-IMC controller.

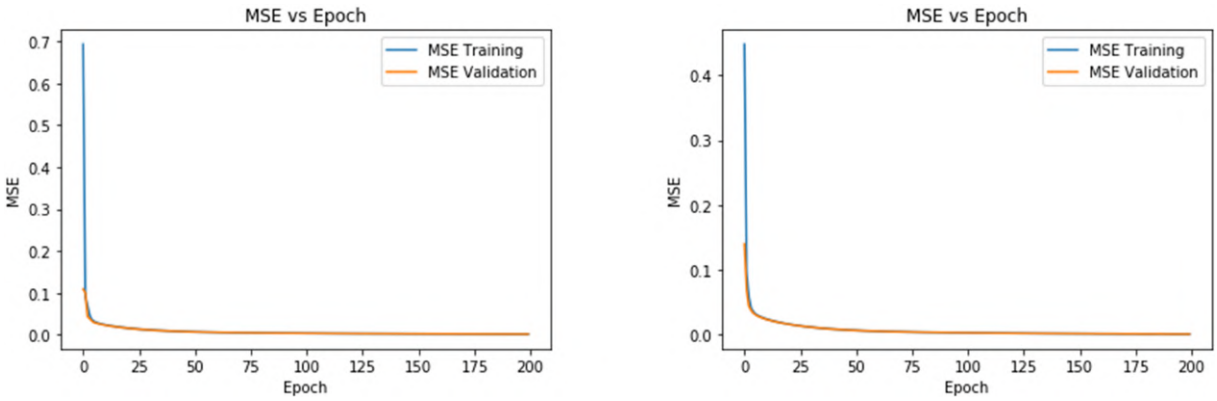


Figure A.2: Learning curves of the ANN_{dir} (left) and the ANN_{inv} (right) of the $S_{NO,2}$ ANN-IMC controller.

A.2 K-fold Cross-validation

Fold	Metric	LSTM direct model			LSTM inverse model		
		Training	Validation	Test	Training	Validation	Test
1st	MAPE [%]	22.839	24.591	24.508	8.766	6.621	4.495
	MAAPE [%]	13.490	13.803	13.896	8.232	8.039	8.437
	RMSE	0.102	0.105	0.103	2.795	2.877	2.937
	NRMSE	0.015	0.016	0.016	0.009	0.009	0.009
	R^2	0.997	0.996	0.997	0.999	0.999	0.999
2nd	MAPE [%]	22.562	23.562	24.837	9.583	4.186	4.382
	MAAPE [%]	13.226	13.802	13.882	8.188	8.137	8.386
	RMSE	0.100	0.102	0.101	2.778	2.772	2.901
	NRMSE	0.015	0.016	0.015	0.009	0.009	0.009
	R^2	0.997	0.997	0.997	0.999	0.999	0.999
3rd	MAPE [%]	22.457	24.310	24.415	9.438	9.293	4.690
	MAAPE [%]	13.327	13.676	13.714	8.437	8.626	8.651
	RMSE	0.101	0.105	0.102	3.091	3.087	3.233
	NRMSE	0.015	0.016	0.015	0.010	0.010	0.010
	R^2	0.997	0.996	0.997	0.998	0.999	0.998
4th	MAPE [%]	22.995	21.738	24.625	9.440	5.501	4.154
	MAAPE [%]	13.729	13.559	14.194	7.921	8.375	8.174
	RMSE	0.103	0.104	0.103	2.697	2.679	2.815
	NRMSE	0.015	0.016	0.016	0.008	0.008	0.009
	R^2	0.997	0.996	0.997	0.999	0.999	0.999
5th	MAPE [%]	23.522	25.734	25.630	9.589	9.257	4.442
	MAAPE [%]	13.597	14.109	14.088	8.197	8.126	8.402
	RMSE	0.106	0.107	0.107	2.842	2.903	2.976
	NRMSE	0.016	0.016	0.016	0.009	0.009	0.010
	R^2	0.996	0.996	0.996	0.999	0.999	0.998
6th	MAPE [%]	24.863	23.380	26.759	10.537	5.176	4.542
	MAAPE [%]	14.028	13.333	14.417	8.349	7.675	8.458
	RMSE	0.105	0.106	0.105	3.005	2.977	3.181
	NRMSE	0.016	0.016	0.016	0.009	0.009	0.010
	R^2	0.996	0.996	0.997	0.999	0.999	0.998
7th	MAPE [%]	24.554	23.308	26.502	9.213	16.412	4.300
	MAAPE [%]	13.818	13.530	14.219	8.087	8.084	8.270
	RMSE	0.105	0.107	0.107	2.790	2.795	2.896
	NRMSE	0.016	0.017	0.016	0.009	0.009	0.009
	R^2	0.996	0.996	0.996	0.999	0.999	0.999
8th	MAPE [%]	23.553	24.344	25.592	10.317	10.903	4.137
	MAAPE [%]	13.594	13.992	14.062	7.832	8.494	8.110
	RMSE	0.103	0.102	0.103	2.680	2.757	2.799
	NRMSE	0.015	0.016	0.016	0.008	0.009	0.009
	R^2	0.996	0.997	0.997	0.999	0.999	0.999
9th	MAPE [%]	21.561	21.001	23.160	9.014	23.148	4.350
	MAAPE [%]	12.621	12.670	13.026	8.127	8.024	8.312
	RMSE	0.094	0.096	0.095	2.912	2.947	3.052
	NRMSE	0.014	0.015	0.015	0.009	0.009	0.010
	R^2	0.997	0.997	0.997	0.999	0.999	0.999

Table A.1: Nine-fold cross-validation performance of the ANN_{dir} and the ANN_{inv} of the $S_{O,5}$ ANN-IMC controller.

Fold	Metric	LSTM direct model			LSTM inverse model		
		Training	Validation	Test	Training	Validation	Test
1st	MAPE [%]	13.346	10.123	6.418	1.321	1.144	0.841
	MAAPE [%]	5.333	4.860	3.641	3.707	3.266	5.297
	RMSE	0.060	0.061	0.063	420.107	415.734	478.204
	NRMSE	0.004	0.004	0.004	0.002	0.002	0.003
	R^2	1.000	1.000	1.000	1.000	1.000	1.000
2nd	MAPE [%]	11.752	11.424	5.435	1.497	1.181	1.156
	MAAPE [%]	5.114	4.802	3.483	3.560	3.458	5.376
	RMSE	0.058	0.057	0.060	390.583	396.605	452.681
	NRMSE	0.004	0.004	0.004	0.002	0.002	0.002
	R^2	1.000	1.000	1.000	1.000	1.000	1.000
3rd	MAPE [%]	12.445	13.589	4.954	1.418	1.470	1.334
	MAAPE [%]	5.102	5.394	3.389	3.762	3.753	5.529
	RMSE	0.053	0.054	0.054	532.322	532.686	571.420
	NRMSE	0.003	0.003	0.003	0.003	0.003	0.003
	R^2	1.000	1.000	1.000	1.000	1.000	1.000
4th	MAPE [%]	11.047	12.227	5.276	1.441	1.010	0.997
	MAAPE [%]	4.926	5.065	3.490	3.513	3.780	5.287
	RMSE	0.059	0.059	0.061	396.571	409.918	441.191
	NRMSE	0.004	0.004	0.004	0.002	0.002	0.002
	R^2	1.000	1.000	1.000	1.000	1.000	1.000
5th	MAPE [%]	11.294	12.797	5.073	2.303	1.624	1.079
	MAAPE [%]	4.813	5.069	3.517	3.821	4.049	5.415
	RMSE	0.058	0.060	0.060	462.432	469.652	508.249
	NRMSE	0.004	0.004	0.004	0.003	0.003	0.003
	R^2	1.000	1.000	1.000	1.000	1.000	1.000
6th	MAPE [%]	12.714	10.991	5.010	1.658	1.328	1.156
	MAAPE [%]	4.773	4.713	3.480	3.603	4.081	5.425
	RMSE	0.055	0.055	0.055	492.024	484.136	546.736
	NRMSE	0.003	0.003	0.003	0.003	0.003	0.003
	R^2	1.000	1.000	1.000	1.000	1.000	1.000
7th	MAPE [%]	8.981	7.677	4.504	1.240	1.433	1.074
	MAAPE [%]	4.294	4.011	2.937	3.638	3.366	5.390
	RMSE	0.054	0.054	0.058	495.765	485.580	546.763
	NRMSE	0.003	0.003	0.004	0.003	0.003	0.003
	R^2	1.000	1.000	1.000	1.000	1.000	1.000
8th	MAPE [%]	11.623	14.312	5.848	2.193	2.680	1.466
	MAAPE [%]	5.109	5.416	3.446	3.845	3.552	5.535
	RMSE	0.061	0.060	0.063	585.951	590.203	639.816
	NRMSE	0.004	0.004	0.004	0.003	0.003	0.004
	R^2	1.000	1.000	1.000	1.000	1.000	1.000
9th	MAPE [%]	11.103	13.467	4.883	1.712	1.149	1.167
	MAAPE [%]	4.744	5.044	3.365	3.582	3.732	5.357
	RMSE	0.054	0.055	0.055	411.578	426.422	479.237
	NRMSE	0.003	0.003	0.003	0.002	0.002	0.003
	R^2	1.000	1.000	1.000	1.000	1.000	1.000

Table A.2: Nine-fold cross-validation performance of the ANN_{dir} and the ANN_{inv} of the $S_{NO,2}$ ANN-IMC controller.