Alma Mater Studiorum · University of Bologna

School of Science

Master's Degree in Computer Science

A DEEP LEARNING MODEL FOR CONTROLLING THE GALVANIZING PROCESS IN A PRODUCTION LINE

Supervisor:
Prof. Andrea Asperti

Co-Supervisor:
Alessandro Ferraiuolo

Presented by:
Salvatore Fiorilla

II Session
Academic Year 2021-2022

"Be thinkers, generate ideas not only data."

Bosh[7] and Nurse[35].

Abstract

In the industrial environment of steelmaking and steel transformation, the process of galvanizing is a chemical treatment which is applied to protect the steel from corrosion. The process consists in coating a steel strip, by dipping it for few seconds inside a molten zinc bath at 450°C. Then, when the strip is pulled out, in order to remove the exceeding zinc, two mechanical nozzles emit a stream of compressed air that hits the surface of the strip. With the air jet, you can arbitrarily clean the exceeding amount of zinc from the strip without any direct contact, and get the zinc coating you want. The effect is commonly known as the air knife effect.

In the scientific literature, the problem of controlling the air knifes is also referred as *wiping process* or *airknife process*. The process can be expressed in the following way:

Given the physical conditions of the process line such as the temperature (t) of the zinc bath, the height (h) of the air blades from the bath, the processing speed (s) and the target value of the zinc coating (c), find the tuple of pressure and distance values (p, d) *such that, by emitting compressed air over the steel strip surfaces with such pressure and from that distance, the air-knife effect attains the target zinc coat (c).*

The study, which was conducted in combination with the R&D department of Marcegaglia SPA, results in a Deep Learning model that is able to drive the air-knife effect. We call it controller. The module takes as input

$< t, h, s, c >$ and generates the expected tuple of values $< p, d >$.

Accordingly to the requirements we designed the structure of the controller network. We collected the data set and we conducted an Exploratory Data Analysis on historical data of the smart factory of Ravenna (IT). Finally, we designed the loss function. The function is the sum of three components: (1) the minimization between the coating addressed by the network and the target value we want to reach; (2) a weighted pressure minimization component and; (3) a weighted distance minimization component;

Our approach requires the construction of a second module, named coating net. It predicts the coating of zinc $< \widehat{c} >$ resulting from the wiping effect when the conditions $< t, h, s, p, d >$ are applied to the production line. We trained and evaluated five different modules. The structure of the most accurate model is made by a linear and a deep component. The linear component reflects an existing mathematical modeling of the process, adjusted, in our cases, by a nonlinear "residual" component learned from empirical observations.

The predictions made by the coating nets are then used as ground truth in the loss function of the controller. By tuning the weights of the different components of the loss function, it is possible to train models with slightly different optimization purposes.

We validated and tested the model according to two families of metrics: the first compares the precision of the current model with the standard one in conditions were the minimum values of pressure and distances were estimations of both modules; the second analyses how the controller modeled the current solutions with the new optimization logic. Tests showed that the new logic can optimize of 50% the sub-optimal values of pressure and of the 20% some distance values; different training strategies can be regularized with a good level of precision: the overall accuracy of the coating target addressed by output values is estimated to be in the range of $\pm$ 3 g/m^2 for all of them; all the other results are presented in this study.

Keywords

Industry 4.0, Hot-Dip Galvanizing Process, Air-knife process, Neural Networks, Deep Learning

Sommario

Nell'ambito industriale della produzione e trasformazione dell'acciaio, il processo di zincatura è un trattamento chimico che serve a proteggere l'acciaio dalla corrosione. Tale processo consiste nel rivestire un nastro di acciaio, immergendolo per pochi secondi dentro una vasca di zinco liquido fuso a 450°C. Quindi, quando il nastro viene tirato fuori dalla vasca un getto di aria in pressione rimuove l'eccesso di zinco senza alcun diretto contatto fisico.

Se controllato propriamente, il getto permette di ripulire in modo arbitrario dalla superfice del nastro la quantità di zinco voluta. L'assottigliamento del rivestimento avviene come fosse un coltello d'aria a farlo. Per questo motivo in gergo tecnico lo si indica come effetto lama d'aria (,in inglese *airknife effect*). In questa tesi presentiamo un modo di controllare l'effetto del getto d'aria in modo ottimale. In letteratura scientifica tale problema è comunemente noto anche come *wiping process* o cone *airknife process.* L'abbiamo formalizzato come segue:

Date le condizioni fisiche della linea di processo come la temperatura (t) del bagno di zinco, l'altezza (h) delle lame d'aria dal bagno, la velocità di lavorazione (s) e il valore target del rivestimento di zinco (c) , trova la tupla dei valori di pressione e distanza (p, d) tale che, emittendo dell'aria compressa sul nastro di acciaio con tale pressione e da quella distanza, l'effetto lama d'aria raggiunga il rivestimento di zinco desiderato (c).

Lo studio, condotto in collaborazione con il dipartimento R&D di Marce-

gaglia SPA, ha prodotto un modello di Deep Learning in grado di guidare l' air-knife effect. Lo abbiamo chiamato *controller*. Il modulo prende come input i parametri di $< t, h, s, c >$ e genera la tupla di valori $< p, d >$ di conseguenza.

Dai requirements abbiamo progettato la struttura della rete. Quindi abbiamo raccolto il dataset e condotto una Exploratory Data Analysis sui dati della smart factory di Ravenna (IT). Infine abbiamo progettato la loss function come somma di tre elementi: (1) la minimizzazione tra il rivestimento indirizzato dalla rete e il valore target che si vuole ottenere; (2) una componente pesata di minimizzazione della pressione e; (3) una componente di minimizzazione della distanza ponderata;

Il nostro approccio richiede la costruzione di un secondo modulo, chiamato *coating net*. Essa stima il rivestimento di zinco $< \widehat{c} >$ risultante dal wiping effect quando le condizioni $< t, h, s, p, d >$ sono applicate alla linea di produzione. Abbiamo valutato cinque diverse architetture al capitolo 4. La struttura di quella piu accurata è fatta da un modello lineare e da un componente deep. Il componente lineare riflette un esistente modellamento matematico del processo, aggiustato, nel nostro caso, da un componente non-lineare "residuale" allenato da osservazioni empiriche.

Le predizioni fatte dalla *coating net* vengono usate come ground truth nella funzione di loss del controller. Facendo tuning dei pesi su differenti componenti della funzione di loss, è possibile addestrare il modello per diverse logiche di ottimizzazione dell'effetto finale.

La validazione del modello è stata applicata su nuovi dati reali confrontando le logiche di tre diversi controllers. I test hanno mostrato che l'ottimizzazione di certi valori sub-optimali può raggiungere il 50% sulla pressione ed il 20% sui valori di distanza. L'accuracy del target di rivestimento indirizzato dal controller invece è nel range di ± 3%. Tutti gli altri risultati e i il comportamento della rete sono stati presentati in questo studio.

Acknowledgments

Foremost, I would like to express my sincere gratitude to my supervisor, Prof. Andrea Asperti, for guided me in this months of work with his experience and knowledge.

My sincere thanks goes also to my co-supervisor: the Doctor in Physics Alessandro Ferraiuolo, for giving me the opportunity to contribute to this exciting project.

For the support I would also like to thank Stefano Orselli very much.

Finally and importantly, I want to extend my most special gratitude to my Family and Sofia for encouraging and supporting me during my university years.

List of Tables

List of Figures

Contents

Chapter 1

Introduction

1.1 Marcegaglia Group

Marcegaglia [1] is an Italian industrial group world leader in the transformation and processing of steel. Today it is the 1st independent operator in the steel processing sector in the world; and 1st manufacturer of stainless-steel welded tubes in the world; 1st manufacturer of carbon steel welded tubes in Europe and; 1st service center in Italy.

The group operates worldwide with 6,600 employees, 60 commercial units and 29 factories (steel mills). It serves over 15,000 customers and works 6.2 million tons of steel annually. Marcegaglia manufacturing activities include flat products and welded tubes, stainless products, drawn bars and heavy plates. The range of products which are made in Marcegaglia, varies from carbon to stainless, from long to flat products, from commodity to specialty. The customers choose Marcegaglia for several applications such as mechanical engineering, industrial plant engineering, construction, automotive and food industry specializations.

The envisions of "4.0 Industry" ([17], [11]) aims at the digitalization of the services and innovation of the processes towards a more "intelligent" and sustainable industrial environment. In this scenario, the R&D department of

[1] https://www.marcegaglia.com/officialwebsite/

Marcegaglia commitment is constant. In the last 10 years, important technological investments have transformed the Marcegaglia plants in Ravenna and Casalmaggiore of one of the largest and futuristic steel service centres in Europe [2]. Ravenna has a plant with a surface area of 540,000 m2, 245,000 of which are covered.

1.2 Thesis outline

This thesis is structured as follows. The first chapter describes the hot-dip galvanizing process in its main steps; a physical focus is done on the air-knife effect: the main study object of the process; furthermore, the chapter presents the formalization of the problem and the related works.

In the chapter 2 the technological background of the concepts concerning Neural Networks and Deep Learning is provided; Residual Networks and Transfer learning are also discussed; the frameworks used, Keras and TensorFlow, are introduced as well.

Chapter 3 provides the methodology approach we used to design our solution; we analyzed the measures, the collected requirements, and the problem of the current logic of control; then we designed the network and the loss function; finally, we collected the dataset and did Exploratory Data Analysis on it.

In chapter 4, we illustrated all the experiments and their results we conducted to project the neural network applied in the the loss: The coating Net.

In chapter 5 we discussed how we trained the model under three different training strategies. We showed what the models learned, and then we tested them in optimization and accuracy. Results are showed in the last section of the chapter.

Chapter 6 draws conclusions and discuss some idea for possible future developments.

[2] https://www.youtube.com/watch?v=dTydGIrowH0&ab_channel=MarcegagliaTV

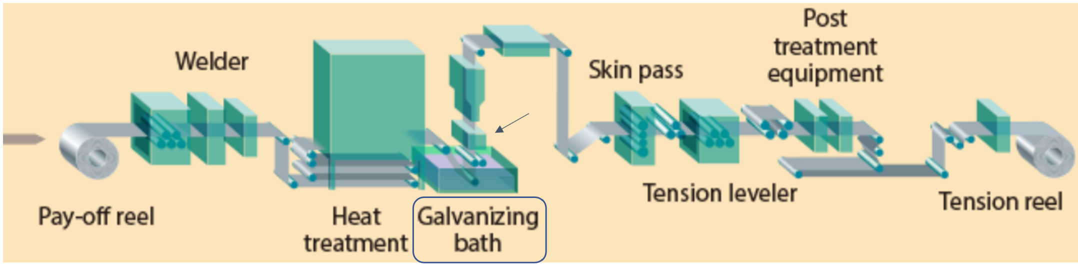


Figure 1.1: Hot-Dip Galvanizing Line.
Source: Marcegaglia R&D.

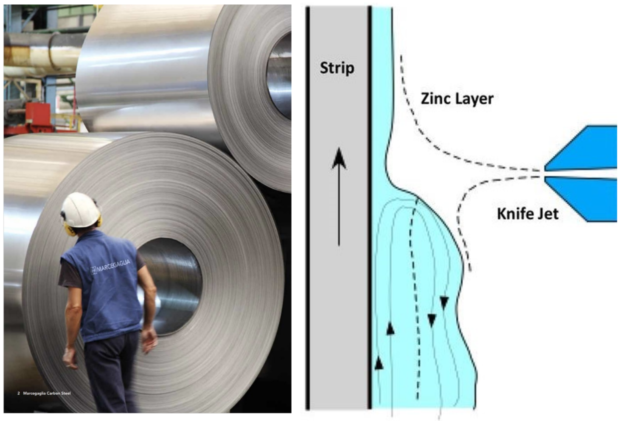
1.3 Hot-Dip Galvanizing Process

In the industrial scenario of steel modelling and transformation, the process of galvanizing is a treatment which is applied to protect the steel from the corrosion. Several methods have been invented by humans throughout the history; they differ from each other in the used material, the thickness and the type of manufacture ([36]). The process, which was studied in this thesis, is the *hot-dip galvanizing process (HDGP)* of a production line owned by Marcegaglia spa. In the figure 1.1 it is showed a typical hot-dip galvanizing line which is similar to the one we focused. The worked objects are large strips of raw steel weighing about 15-25 tons: see img. 1.2, fig. (a).

When the raw steel rolls arrive into the factory, they are released in a dedicated area until they enter in the process line. It is not a matter the length of the strips because the process line works continuously. When a new steel sheet is ready to be processed, its front part is welded with the end of the previous one and the two strips are joined together.

In a typical hot-dip-galvanizing line, at the beginning of the process the steel strips is washed and cleaned with sponges and water; subsequently, the product is annealed in a furnace to check the status of its physical and chemical characteristics.

After the sanity check, the coating process occurs: the raw steel strip is dipped into a molten zinc bath at the temperature around 449 °C (840 °F) for few seconds (fig 1.1: galvanizing bath step). The pure zinc (Zn) reacts



(a) (b)

Figure 1.2: (a) Steel strip. (b) Air knife effect.
Source: Marcegaglia R&D.

with oxygen (O2) of the atmosphere and forms zinc oxide (ZnO). This latter further reacts with carbon dioxide (CO2) and form zinc carbonate (ZnCO3). The molten zinc that is get is usually dull grey, and can protect the steel underneath from further corrosion ([13],[31]).

When the strip is pulled out there are always zinc residues on the surface of the object. In order to remove them, two mechanical nozzles are installed just above the zinc bath and close to the liquid level. They are positioned one in front of the other and the strip scrolls upwards in the middle of them. The arrow in fig 1.1 indicates their exact position in the line. The sides are technically called *Top* and *Bottom*. By emitting a powerful and consistent stream of air, you can arbitrarily clean with any direct contact, the exceeding amount of zinc from the surfaces of the strip and get the zinc coating you want. *air-knife effect* ([26]): img. 1.2 fig. (b) show it in detail.

The process ends by passing the strip through a series of auxiliary processes, for instance: the skin-pass e tension leveler are two steps in the process that modify the surface structure of the strip to make it suitable for post-treatment operations such as deformations or painting ([15]). Other steps are also cooling the product and quality control.

Finally, the galvanized steel strip is cut by shears, re-rolled and placed

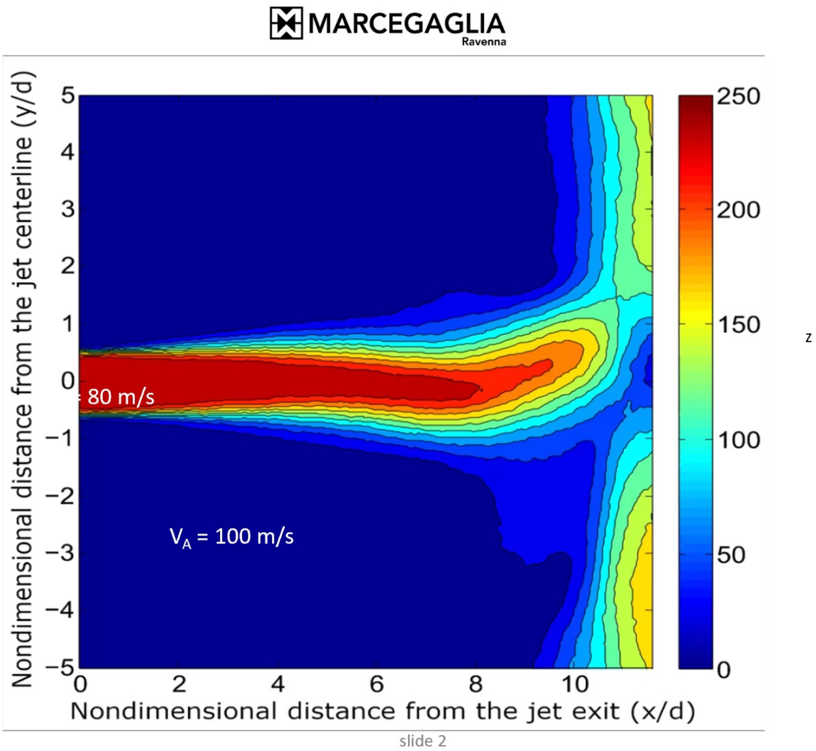


Figure 1.3: Motion of the air stream
Source: Marcegaglia R&D.

through mechanical cranes in the storage area.

1.4 Wiping effect: a physical point of view

This section provides an overview of the physical phenomenon that occurs during the *air knife effect*. It is helpful for understanding the requirements that will be discussed at chapter 3.

In the galvanizing process, the term refers to the effect which is obtained at the exact step of cleaning the excess zinc from the surface of the steel strip. It is depicted in the image 1.2 fig (b); the effect is applied to both sides of the strip and resembles a knife, made of air, that literally cuts grams of zinc.

From a physical low-level point of view, like the one that is depicted in the figure 1.3, the effect is not so stable. When the air is sprayed on the environment it reaches the surfaces with a trajectory that is *chaotic*; to better convey the idea, in the slowest and most controlled case it may be comparable to a sinusoidal wave. The disorder and confusion of the trajectory increase with increasing distance and working speed. When this happens, to have a stable *"knife"* and maintain the same weights of zinc in the surfaces, many more millibars of air pressure are required.

The quality of the final product is strictly related to this phenomenon. If the pressure is too high, the surfaces can be less uniform and much rougher. Cracks and wrinkles can arise.

It should be clear now that the problem is about finding the right (sufficient low), pressure level to apply at the minimum distance. The combination of the two factors reduces the chaos and leads to a better final product and, in the time, to lower electricity consumption.

1.5 Problem Description

Our research envisions a software solution to control the mechanical nozzles of the line and, as a consequence, the air-knives effect applied to the surface of the steel strip. In the scientific literature, this problem is also referred as *wiping problem* or *air-knife problem.* We formalized it as follows.

Given the physical conditions of the process line such as the temperature (t) of the zinc bath, the height (h) of the air jets from the bath, the processing speed (s) and the target value of the zinc coating (c), find the tuple of pressure and distance values (p, d) such that, by emitting compressed air over the steel strip surfaces with such pressure and from that distance, the air-knife effect attains the target zinc coat (c).

The multi-output regression problem presents the following challenges:

1. All the values are continuous; for the same input, several tuple $< p, d >$ can satisfy the same coating target; for instance: at the same speed, temperature and heights, the coating of zinc of 52 g/m^2, can be obtained by the tuples:

   - $< p = 160(mBars), d = 7.5(mm) >$
   - $< p = 250(mBars), d = 10(mm) >$.

2. The computation of the results has to take into account the phenomenon of the chaotic motion of the air in some way. We see it in the previous sections from a phisic point of view.

3. Each strip presents unique unmeasured characteristics such as the tension and quantity of its alloy components; they influence in some way the coating of zinc; this consideration is also carried out in [20].

4. The metrics of the process are measured by sensors that are installed along the components of the production line. Some measurements may not be perfect; certain measurements must necessarily be taken many meters ahead of the moment in which the real effect occurs. The reasons are due to physical impossibilities or because the sensors were designed for other purposes. For instance: when the coating of zinc is cut out by the air, it is still very hot in that moment and, it can be measured with the current technology. So, the activity occurs when the strip is cooled down. As a consequence, there is always a probability that some bias in measurements are generated.

The idea, which was proposed by the R&D team, is to apply the neural network in the controlling logic of the air knives and try to address the four challenges. We have devised a deep learning model which is able to drive appropriately the air-knife effect; we call it controller. The module takes as input the value of $< t, h, s, c >$ and generates the tuple of values <p, d>accordingly to the requirements and constraints of the process. We discussed them, the design and the methodology of our study in the chapter 3.

1.6 Related Works

Increasing the resistance to the corrosion and, providing a better look of the surface of the end product, are the two general improvements which engineers are looking for in the hot-dip galvanizing process. The two research fields are commonly accepted by the scientific literature. [57] presented a review of the development of the anti corrosion processes; the emerging coating protective systems are duplex-technique based. In their work the authors concluded also that the research filed needs an unified lifetime evaluation methods for such coating systems.

Our study focused on the other side of the challenge: to improve the external surface of steel objects in beauty. The challenges to face are the look defects which can arise from galvanizing process; i.e.: bare spot, distortion, dross, touch marks, dents, rough coating, ash deposit, reactive and non-reactive steels welded together. The remedies can involve the correct management of the entire process, e.g., the handling of the temperature of the zinc in the pot; or can also involve a general development of new technologies ([23]): in [47] advanced multi-slot pumps are presented and in [24] a feedback-based system to reduce the vibration of speed fluctuations in the lines.

Furthermore, the correct and accurate application of the tools is another form of developing of new technologies. It can obviously lead to better surfaces as well. The study presented in this final thesis focused on a software solution which was modelled to control the mechanical pumps of the process in an optimized way.

In this section, related works are presented.

Machine learning (ML) for process control in industry is a concept that is gaining popularity ([6]); two questions addressed our research; we asked ourselves: (1) Are there machine learning models applied to the hot dip galvanizing process? (2) What are the standard solutions applied today for controlling the hot dip galvanizing process? Do any of them use deep learning? We answered in the same order.

Guelto *et al.*in [20], recognized the utility of ML trying to predict the weights of the coating; in the work the authors defined this problem nonlinear and they also recognized that there is unknown number of unmeasured variables which influence in some way the coating of zinc. Such variables reduce the prediction of the zinc. A similar analysis was made also in [18]: the purpose of this article was to apply data mining techniques such as *NN*[3], *SVM*[4], *SVM*[5], and *RT*[6], to predict the mechanical properties of the galvanized steel. According to their results neural networks were the much accurate. Also [12] did a comparative study; It was made between several types of algorithms of AI for the design of a prediction model that will allow to determine the strip's variation in temperature. MLP neural networks over perform other Data mining Techniques. Then, it pointed also out that the model are not suitable for predicting the behavior of strips of hot pickled coils because their surface conditions are substantially different. The authors of [40] have the same point of view about the utility of neural network models in the process line context; in their article of last year, they applied the NNs in combination with a multi-objective genetic algorithm to predict mechanical properties of a line such as: the yield strength (YS [48]); the ultimate tensile strength (UTS [9]); and the elongation at fracture (EL [50]).

Finally, examples of application of neural network in the design of online process control cycles are [42] and [37]: [42] integrated the NNs in the online temperature management of a continuous annealing furnace on an HDGL. In the other article, [37], the authors developed a neural network model to

[3] Neural Networks
[4] Support Vector Machine ([3])
[5] Regression Analysis ([4])
[6] Regression Tree Methods ([10])

effectively design the process cycles for online control of a battery of batch annealing furnaces.

Let's diving deep now into the question number (2). Actually, from our research, we did not found many different approaches. So far, manual approach is very used: operators push air for a while and wait for the feedback of coating.

Three papers are addressing the problem of the coating weights control with *first principle* [7] methods; two of them are really old: [49] was published in 1976 and [52] in 1996; we decided to let them in this study just as historical references. A much recent work that use the first principle to derive a mathematical model is [14] (2007): such model describes the pressure and wall shear stress distribution. It is applied in combination with other techniques, to predict the coating weights from the process parameters. The test reported good results but, however, the online control of the weights looks complex. Conversely, the much used method is definitely *Linear Regression*([25]): [46] proposes a model of two components (math formulas); one to predict the pressure and distance of the air knife from the strip; and the other one, it is used to handle the linear change which occurs in steady state. [20] is another similar example of linear regression too. Both methods [20] and [46] are exponential predictors whose parameters were get by different optimization algorithm over historical process data. They have the big advantages that can be easily realized. However, as linear regressions, they don't generalize well for the non linear features of the system.

In 2016, the same author of before: Guelton, in [21] published a work for the ArcelorMittal's hot dip galvanizing line at Florange; it was the extension of the previuos one; They improved the system, which before was just a formula, by adding a new features: the electromagnetic strip stabilization.

However, the same problem: including the flatness of the strip in the control method, was solved in [43] with no electromagnetic equipment. And this helped the process line that were not equipped with it.

[7] https://en.wikipedia.org/wiki/First_principle

Finally, in [16] an advanced linear regression model controller is proposed: the model checks if the speed proposed by the metallurgical and furnace models allows to reach the target zinc thickness; according to the target zinc thickness and line speed, their equation estimates distance, pressure and height of the air knives. Then they control the pressure and distance using a PID [2] regulation loop. They showed performance is of the 2% of nominal target in steady state conditions. The model considers also the tilting of the strip in its equation by projecting the value in the the linear regression formula as the angle of the nozzles.

The latest approach for controllers we found are the *Neural Networks*:

In 2017, [38] used a NN to predict the coating in an optic of developing an effective coating weight control system. In their results they showed an accurate model which performed well when the coating as to increase and lose something in accuracy when the coating have to decrease and there is a speed decrement too. By the way, the logic of pressure and distance is embedded in the workflow of the project and, hence, sometimes may not be optimal.

Another example of a similar application of Neural Network is in [32]. The network and the Genetic Algorithm ([54]) are used to modelling a coating thickness predictor and put it into the system. However, there is no controlling logic at all, just prediction.

The work which is much similar to the one of this thesis is a novel neural networks control system which were proposed by [45]. They used a loss of minimization that is different of our loss function. They take into account only the pressure and minimize it by using constant normalized values. Distance minimization is not considerate at all for what they presented. Secondly, their application logic is embedded in the process and concerns many predictors: they distinguish the state of the application between speed and coating changes. Conversely, our study manages all this changes into one single module.

Chapter 2

Technological Background

Many subjects of *artificial intelligence (AI)* mostly concerns functions that are easy for humans to compute and, at the same time, difficult to describe in terms of elementary operation. The overall *Machine Learning (ML)* idea is to study any set of parameterized functions as function approximation. We focused on the specific set of composable function called neural networks. In this section we are going to provide the technological background which the reader reserves to understand the concept we adopted in this thesis.

2.1 Neural Networks

A *Neural Network (NN)* is a recipe for computing a function. It is made of Neurons organized in parallel into layers. Each neuron is itself a very simple function: signals that income into it are weighted and added together; Then, the they are *fired* by comparing the value of that sum against some threshold. *Deep Neural Networks(DNN)* are those nets which are composed of multiple layers in sequence. The parameters of a deep neural network are composed by the firing thresholds, the weighted connections between the neurons and their bias. The state-of-the-art neural networks can have over 100 billion parameters; an example is [8].

A neural network can be seen as a parametrized function:

$$f(x; \theta) \tag{2.1}$$

It takes in input a vector x of values to compute; and a large vector of parameters θ which determines its shape. We initialize the network by randomly sampling the parameter vector θ from a computationally simple probability distribution,

$$p(\theta) \tag{2.2}$$

The initialization distribution induces a distribution on the network output. If it is random at first, the network output must also be random. Then, we tune the high-dimensional parameter vector as $\theta \rightarrow \hat{\theta}$, such that the resulting *network function* $f(x; \hat{\theta})$ is as close as possible to a *desired target function* $f(x)$:

$$f(x; \hat{\theta}) \approx f(x) \tag{2.3}$$

This activity is called *function approximation.* We can tune judiciously $\hat{\theta}$ by fitting the network function $f(x; \theta)$ to a set of many observed pairs of tuples $< x, f(x) >$. Such set is called *training data*; the procedure of making adjustments to the parameters *training procedure*; the algorithm used to perform such procedure *learning algorithm* ([41]).

The basic unit of a layer, the *neuron*, is also called *perceptron.* It's structure is showed into the images 2.1. It consists in two operations:

- The *preactivation* phase z_i computes the linear regression formula

  $$z_i(x) = (\sum_{i=1}^{n} w_i x_i) + b \tag{2.4}$$

  over the incoming input $x = (x_1, ..., x_n)$.

- Then, the "evidence" of weights and bias: i.e. the z_i value of output, is applies to a non-linear and derivable *activation function* : φ .

  Examples of activation functions are: sigmoid, tanh, sin, linear, ReLu, softplus, swish, gelu. The perceptron formula of a single neurons can

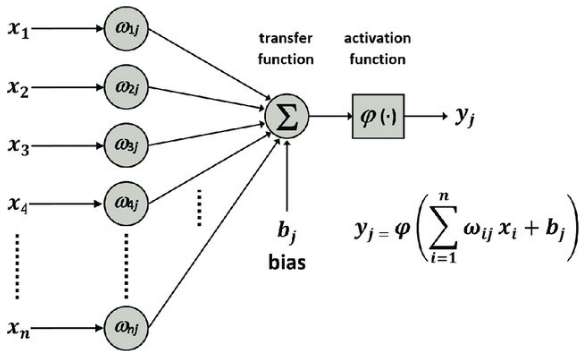


Figure 2.1: Scheme of a perceptron [1].

be rewritten as

$$y_i = \varphi(z_i(x)) \tag{2.5}$$

and the output of the n-nodes layer ℓ is a vector

$$z^\ell =< y_i, \dots, y_n > \tag{2.6}$$

There are two *learning algorithms* to train a neural network module: one way is to randomly perturb the network parameters and see if we get better results; This approach is called evolutionary; It has a high probability of making worse predictions;An examples is in ([56]). The second way is the back-propagation algorithm; it provides instantiation of the gradient descent method to train deep neural networks ([5]).

2.2 Deep Learning Expressiveness

From the point of view of the expressiveness, a Neural Network (NN) with at least two layers (shallow) and an appropriate number of neurons, can arbitrarily approximate each function $f : \mathbb{R} \to [0, 1]$. Moreover, by adding hidden layers to the network you can have the same functions approximation with a lower number of neurons.

Every time a layer is added, is performed a combination of non-linear functions such as

$$(g(f(...z(x)..)) \tag{2.7}$$

Hence, the NN is a chain of non linear functions. The non linearity introduced into the network by the function is what allow to approximate arbitrarily complex functions. In other words, if you use linear activation functions in the chain, adding layers to the network doesn't gain computational power because composing linear functions always results in a linear function. No matter the network size.

Actually, there is no deep learning without neural network. Hand engineering features are time consuming, brittle, difficult to get and not scalable in practice. The big advantage of deep learning is the ability to extract automatically low, mid and high level patterns from texts, images, videos, sounds, numbers and even graphs and series of data. In particular, the layers closest to the top of the network topology stack, are able to obtain high view patterns. For example, in the case of images processing you can recognize a nose or an eye, or in the case of sound the musical genre (contents taken from :[19] and [58]).

This is why DL is not only a branch of ML or AI but it is also the hottest research field for advanced computer science topics such as Natural Language Processing ([27]), Computer Vision ([34]) or Robotics ([29]).

2.3 Residual Networks

We can modify the deep network architecture so that the hidden layers only have to learn a residual function. The standard way of the generic non linear $(\ell + 1)$-th layer equation is

$$z^{(\ell+1)} = L(z^{\ell}; \theta^{(\ell+1)}) \tag{2.8}$$

Where L is the function of the layer $\ell + 1$; z^{ℓ} is the output of the previous layer: equation (2.6); the parameters in $\theta^{(\ell+1)}$ define the shape of L.

In place of it, we can design the layer as

$$z^{(\ell+1)} = F(z^{\ell}; \theta^{(\ell+1)}) + z^{\ell} \tag{2.9}$$

such that the *residual block* F is the residual mapping of the function we want our layer to learn. The equation 2.9 doesn't introduce extra hyper-parameters compared to 2.8; the residual block F often has two or three layers; furthermore, for the definition of 2.9 the dimensions of x and F must be equal.

The approach helps to reduce the vanishing gradient problem [1] which occur when the parameters of the network are close to zero. The pre-activation of z^{ℓ} in the layer propagates an *undegraded* copy of the input signal and importantly, gains in accuracy of the results. This helped in training very deep models such as ResNet [22]. It has been empirically shown that residual blocks lead to a significant increase in test performance even for the deepest networks.(Appendix B of [41] and [22]).

2.4 Transfer Learning

The idea of *transfer learning* is that a generic representation of non-linear features can be learned by a network during the training; there is evidence that the layers of a trained *CNN* can map the characteristics of an input with increasing complexity with the level of *"depth"*; lower levels can likely act as predictors of patterns independent of the specific type of activity; such *learned knowledge* can be transferred from existing tasks to new one ([51]): the parameters of the pre-trained network can be injected inside the layer of another module; this helps the base network to perform the new tasks better. To *transfer the learned knowledge*,the input data of the two models must be the same. Pre-training is often done on a dataset larger than the target one.

With the new initialization the model can train all or part of its layers by freezing them. This allows to "fine-tune" the higher-order feature representations of the base model ([39]).

Given $\mathcal{D}$ a specific domain defined as:

$$\mathcal{D} = \{\mathcal{X}, P(X)\} \tag{2.10}$$

[1]https://en.wikipedia.org/wiki/Vanishing_gradient_problem

Where $\mathcal{X}$ is a feature space[2] and $P(X)$ a marginal probability distribution [3], with $X = \{x_1, ..., x_n\} \in \mathcal{X}$. And given also a task $\mathcal{T}$ which consists of two components too:

$$\mathcal{T} = \{\mathcal{Y}, f(x)\} \tag{2.11}$$

where $\mathcal{Y}$ is the label space and $f : \mathcal{X} \rightarrow \mathcal{Y}$ is the objective predictive function. Such tasks $\mathcal{T}$ is learned from pairs of $\{x_i, y_i\}$ of the training set where $x_i \in X$; $y_i \in \mathcal{Y}$ and $y_i = f(x_i)$.

If we define two different domain and tasks, source (s) and target (t); which differ each other: $\mathcal{D}_S \neq \mathcal{D}_T$ and $\mathcal{T}_S \neq \mathcal{T}_T$; *transfert learning* helps to improve the learning of the target predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ (see [28]).

2.5 Tensorflow and Keras

We implemented the network by using Tensorflow [55] and Keras [30] frameworks. TensorFlow is an end-to-end, open-source machine learning platform which provides also an infrastructure layer for the differentiable programming. It can compute efficient low-level numeric computation of tensor object on CPU, GPU, or TPU. To have a better idea of what TensorFlow is appropriate to do, we can address the reader to the most famous automatic differentiation algorithms: the computing of the gradient of an arbitrary differentiable exspression [4].

Furthermore, a TensorFlow program supports the Eager execution (a program is described a graph); Distributed and Scalable processing and Cross-Platform. In TensorFlow Graphs are data structures that contain a set of *Operation and tensor objects*: operation are units of computation and tensors are the units of data that flow between operations. Both of them are defined in a *Graph context*. Since these graphs are data structures, they can

[2]The vector space of an n-dimensional vector which is made of numerical features of an object.

[3]It is the probability distribution of the random variables contained in the subset.

[4]https://www.tensorflow.org/guide/autodiff

be saved, run, and restored all without the original Python code.

Keras is the official high-level API of TensorFlow 2: an interface which provides abstractions and building blocks for developing and shipping machine and deep learning solutions. Keras empowers engineers and researchers to take advantage of the capabilities of Tensorlfow 2: you can run Keras on TPU or on large clusters of GPUs, and you can export your Keras models to run in the browser or on a mobile device.

Chapter 3

Methodology

In this chapter the methodology of our work is presented. We analyzed the measurements of the smart production line and we collected the requirements. Then, we addressed both into network and loss function designing sections. The design activity was the guide for all subsequent experiments presented in Chapters 4 and 5. Finally, we collected training and test sets from historical data of Marcegaglia production. We explored data and we gained insights over the features.

3.1 Process measurements

Sensors are installed in the smart process line of Marcegaglia; they constantly measure the steps of the galvanizing process. In this section the measures of our interest will be described; figure 3.1 gives a 2D view of the main phase: the emergence of the strip from the zinc bath and the coating definition. The image also contains the sampled measurements.

Such metrics are:

- (p): The pressure of the air which is streamed by the two nozzles. It is a measures of millibars.

- (d): The centimeters of distance of the nozzles from the surface of the

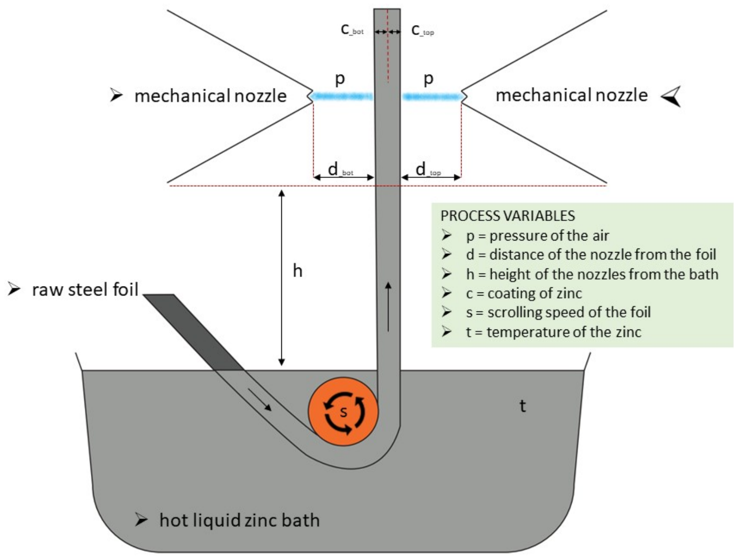


Figure 3.1: Galvanizing process measurements

strip. Two values are taken, one for each side. Typically in the process they are named *top* and *bottom* side.

- (c): The parameter represents the quantity of grams of zinc coating in a squared meter (g/m^2) for each face: *top* and *bottom*, from the theoretical center of the steel strip.

- (h): It measures the millimeters of heights of the nozzles from the bath.

- (s): The speed of working is taken as meters of strip scrolling per minutes (m/m). It changes for instance, when the production in a day has to be increased.

- (t): It is the temperature in Kelvins of liquid zinc inside the bath.

According to the approach of the problem we addressed, the steel strip is always in the middle of the nozzles. The jet pressure is the same for each nozzle and the blowing systems move symmetrically. The same zinc coating can be obtained at high speeds by increasing the pressure and reducing the distance.

3.2 Requirements

In this section we collected the requirements we needed to define the behavior of the network and which are the reasons they had to be addressed.

The neural network must control *the air knife effect*; the context variables of the production line and the desired galvanization grams are taken as input; the model must predict in output the distance of the nozzles from the surface and the pressure they flow to reach the coating target.
We called the network **Controller**.

Input can be received by both humans or external events: operators can change the speed of working or the target of the coating; the temperature of the zinc inside the bath is not always the same in a day; and nozzles can change heights due to vibrations. All this kinds of changes impact the final product production and the network should also learn to control them.

Furthermore, the *controller* should also guide the nozzles in order to achieve the standard average accuracy of the coating target: $\pm$ 3 $grams/m^2$ of zinc on each side of the strip.

Finally, to satisfy the remaining two requirements, the *controller* must also behave in the following ways. (1) It must keep both output values low: the zinc surfaces are uniform when the pressure is lower; small distance values reduce the chaotic movement of the spreading air. The phenomenon was described in section 1.4.

(2) There are lower limits for the output values that must be respected; they are safety limits which come from regulations: the value of the minimum distance possible is fixed at 7.5 cm; the minimum pressure level of the air jet is 160 or 250 millibars depending on the processing speed.

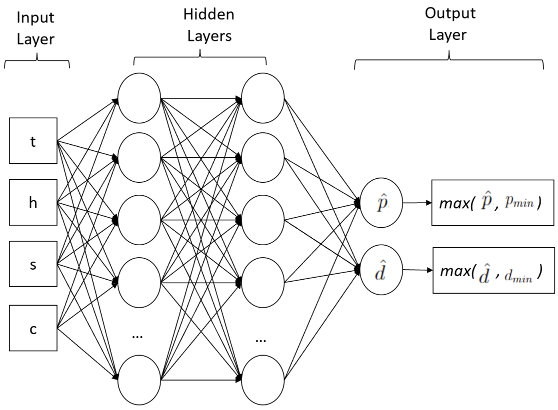


Figure 3.2: Controller Net Structure

3.3 Lacks of the current control system

Before to continue with the design evaluation of the network, we should briefly explain in what the current control system leaks. The hot dip galvanizing production line is driven by the state of the art of the controlling systems of the production line. It is able to perform very accurate values of pressure and distance prediction: we evaluated its accuracy in the latest experiment we carried in chapter 4.

Despite the good results, the current logic does not take into account the two requirements that we introduced previously: low values in air jet and nozzle-stripe gap; and respected safety limits of pressure and distance. The first one produces general better surfaces; the second guarantees the safety for the workers and also avoids possible damages of the production line components.

The *controller network* we envisioned should provide a new logic of solutions $< \hat{p}, \hat{d} >$ which can never be met with the current one; they should handle the non-linear features of the process lines and respect both additionally requirements.

3.4 Controller Network Design

The structure of the network we evaluated is showed in figure 3.2. The parameters of input are $< t, h, s, c >$: the c value is intended to be the target coating to reach, and the other parameters of the tuple are the working conditions of the process line at some moment. The output of the network is the tuple of values $< \hat{p}, \hat{d} >$. The topology consists of two dense layers of 64 units each and *swish* is the activation function we applied.

The network has to be trained to address pressure and distance value combinations to produce the coating target if applied. Furthermore, to satisfy the requirement (1): the network should also learn to choose, among all possible values $< \hat{p}, \hat{d} >$, the tuple which minimizes both. The explanation of the loss function designing is provided in the next section. Finally, we can meet the requirements (2): respecting the lower bound values, by clipping the output with the application of the *max* function. In this way we forced the network to produce only values inside the allowed range. To cut the pressure values out of range, we look at the speed input parameter (s). Distance clipping, on the other hand, occurs with a constant value.

3.5 Loss Design

For addressing the network towards the planned behaviour of the previous section, we designed our ad-hoc loss.

The function of loss needs a way to address the resulting coating from the application of the output values in the process line; it may be, for example, the prediction of a math formula or another neural network; with such accurate predictions then, we can guide the network towards appropriate solutions by minimizing the distance between the predicted coating, and the target coating to be achieved.

In our approach we set out to train from empirical observations another neural network; and then we evaluate with it the coating resulting from the application of $< \hat{p}, \hat{d} >$; we called it **Coating net**. The model takes in input the values $< t, h, s, p, d >$ and estimates the corresponding coating value $< \hat{c} >$. All the details of such network and how we built it are described in the next chapter.

In addition, in order to make the *controller* able of satisfying the other behavioural requirement, we made also a second observation about the loss; it was, to reduced each values of the tuple by minimizing the distance of both outputs from their minimum bounds.

The designed total loss is the sum of three elements: (1) the minimization between the target coating and the coating which is get by applying the generated tuple $< \hat{p}, \hat{d} >$; (2) a weighted pressure minimization component and; (3) a weighted distance minimization component;

We reported the math formula below:

$$Loss_{\varepsilon}(< \hat{p}, \hat{d} >) = mse(c, \hat{c}) + w_1 * rmse(\hat{p}, p_{min}) + w_2 * rmse(\hat{d}, d_{min}) \quad (3.1)$$

where:

- the tuple $< \hat{p}, \hat{d} >$ is the output of the network.
- c is the input parameter of to the controller and the target coating to be achieved.
- $\hat{c}$ is the output of the coating net.
- p_{min} is the minimum pressure. It can vary with speed of input.
- d_{min} is the minimum distance, a constant value (7.5 cm).
- w_1, w_2 are weights to define the influence of the components in the total loss. They should be tuned properly. In chapter 5, we explored them.

3.6 Loss computation algorithm

In this section we are going to illustrate the steps of the loss computation algorithm we used to train the controllers. It occurs in five steps:

1. Once the $< \hat{p}, \hat{d} >$ are generated from the network, they are combined with the input parameters in the tuple $< t, h, v, \hat{p}, \hat{d} >$; such tuple is given in input to the coating net; the obtained output $\hat{c}$ is a coating estimation that the target tuple $< \hat{p}, \hat{d} >$ should get if applied.

2. We measured the *means square error (mse)* between such estimation with the coating target c.

3. The second step concerns the computing of the *root mean squared error (rmse)* between the generated value of pressure $\hat{p}$ and the minimum pressure p_{min}. This last one changes at runtime according to the speed of input. The obtained rmse is multiplied by a weight w_1 to define the intensity of the component over the total loss.

4. The *rmse* between the $\hat{d}$ and d_{min} is computed and multiplied by the weight w_2.

5. We sum up the three losses as showed in the formula 3.1.

It is intuitive to add that the designed overall approach can be generalized for any such multi-output regression problem. We adopted it to model the network output towards a logic of optimized control solutions.

3.7 The Dataset

According to the problem requirements and the network designing, we gathered the dataset from the smart process line data of Marcegaglia. In the table 3.1 we reported the features description and their id; since we faced the problem as symmetric, we also added two more columns: "c" and "d"; they are calculated by averaging the measurements of the top and bottom faces for the distance and coating metrics. The size of the training set is 6364 records, the test set contains 1901 samples. Data were pre-processed into logarithmic scale before to be fed into each networks; the metrics and error losses were always calculated in normal scale values.

Features of the Dataset			
Features	Variable/ID	Description	Units
Temperature	t	of the liquid zinc in the bath	kelvin(abs)
Height	h	of the nozzles from the zinc bath	millimeters
Speed	s	sliding of the strip	meters/minute
Pressure	p	is taken in the nozzle	millibars
Avg Coating *added*	c	$(c_{top} + c_{bot})/2$	grams in m^2
Avg Distance *added*	d	$(d_{top} + d_{bot})/2$	centimeters

Table 3.1: Dataset Features description

Measures	t	h	s	c	p	d
μ	483	650	91	89	268	14
σ	11.09	0.26	25.56	37.80	120.38	4.7

Table 3.2: Features distribution

3.8 Exploratory Data Analysis

We investigated over the dataset. According to our exploratory data analysis the collected data are valuable, accurate, complete and consistent; the probability of get an anomaly or missing data in the values is very low; the human error probability is also close to zero. In the table 3.2 we described the data distribution for each features; The following two sections introduce the Pearson correlation matrix and lead to other final considerations of the *Exploratory Data Analysis* ([44]).

3.8.1 Pearson's correlation matrix

The *Pearson' correlation coefficient* is used to measure linear correlation between two variables. It is the ratio between the covariance of two variables and the product of their standard deviations; such ratio normalize the measurement of the covariance in a range of $[-1, 1]$; when a value is close to ± 1, it indicates a perfect degree of association between the two variables; as the coefficient goes towards 0, the relationship between the two variables will be weaker; the direction, positive or negative, of the relationship is indicated by the sign of the coefficient;

However, the measures can only reflect a linear correlation of variables, and ignores other types of relationships or correlations. The *Pearson's correlation matrix* collects *Pearson coefficients* for all variable' pairs that we want to analyze; we used it in the next section ([53]).

3.8.2 Exploratory Data Analysis Results

A 5D chart is difficult to analyze. We preferred to show the *Pearson coefficient matrix* to highlight the linear relationships between the features: see figure 3.3.

The insights are the following:

- distance and speed are directly proportional to the coating of zinc over the surface;
- if the pressure increases the zinc decreases;
- height appears to have no linear relationship; the reason is that the values are fairly constant; This is also shown in the shape of the table 3.2.

  However, information can be retrieved from this parameter: to make an examples, we introduced into the matrix the value *H_AbsDev*; it refers to the absolute deviation of the height values from the mean; the *Pearson's coefficient* captured a positive linear correlation with distance (+0.12) and a negative correlation (-0.12) with speed.
- temperature has no linear correlation with other features too. By the way, we considered to maintain such features because the parameter can describe the the density of the zinc.

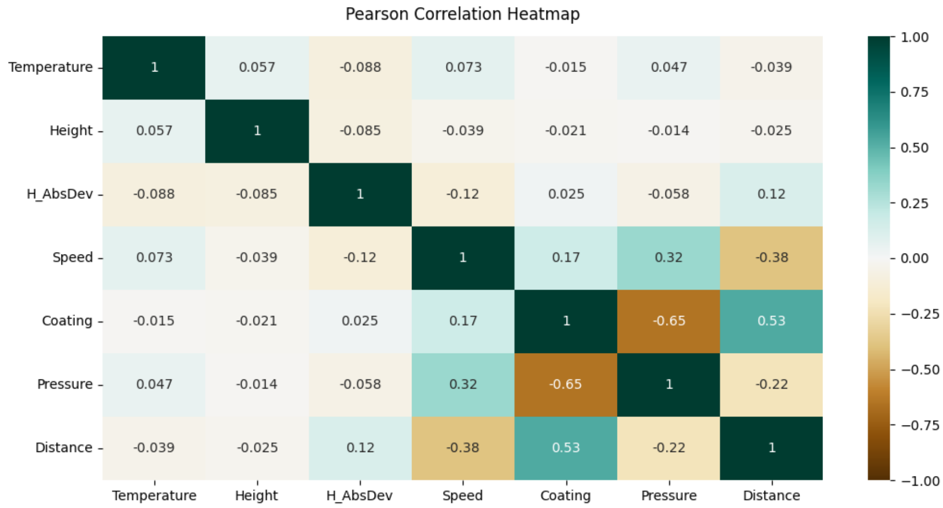


Figure 3.3: Pearson Correlation Matrix

In addition to the values of the *Pearson Correlation Matrix*, we have reported the *pair plot matrix*: figure 3.4. In the color scale of the plots we put different coating measures: dark colors mean an higher coating; the combinations *Speed-Distance-Coating* or *Pressure-Temperature-Coating* immediately strike the eye. Another important consideration we worth to add is that: if the coating has to decrease and speed decreases too, pressure and distance decrements have to take into consideration the quantity of decrement of the speed; this pattern can be found in the pair plot too by seeing the combinations of the plots of the speed with distances and pressures. This is also what we would like our *controller* to learn.

3.9 K-fold Cross Validation technique

We planned in this phase also a series of experiments which we conducted in the next chapter; we trained different possible predictor of *Coating Net* by using the *5-Fold Cross Validation* technique; the method avoids over-fitting and ensures much robust metrics evaluation.

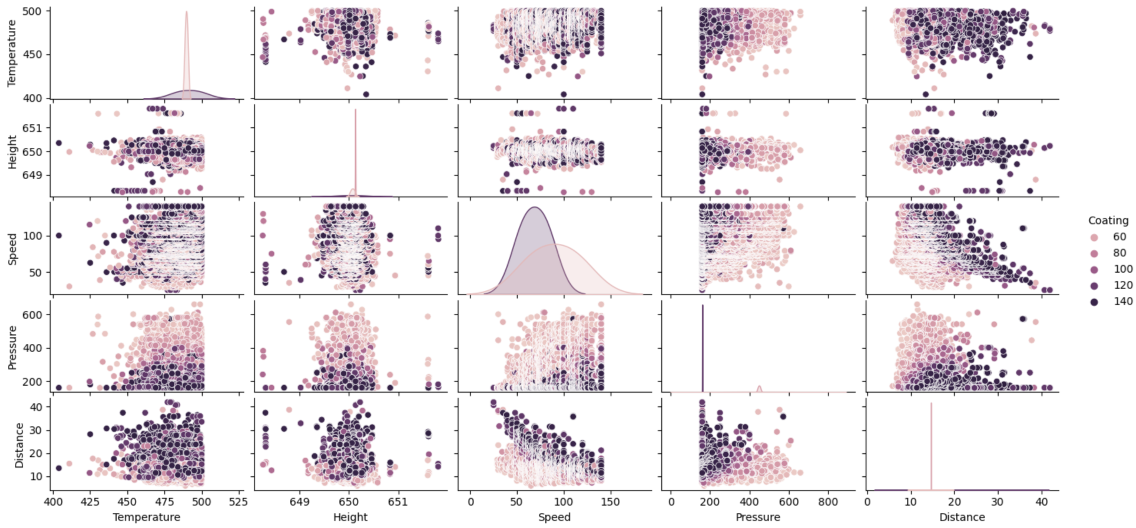


Figure 3.4: Pair plot Correlation Matrix

Cross-validation (CV for short), is a statistical method which is common practice to apply to compare the performance of (supervised) machine learning models; *k-folds* term refers to the usage of the re-sampling technique applied in training phase; that's why the entire procedure is often called *k-fold cross validation*.

In the initial phase, the training set is shuffled and then it is split into k smaller sets called *folds*; subsequently, different models are trained by using, in iterative way, *k-1* folds as training data and the *k-th* fold as validation set. Lastly, the models are tested over a set of data which was hold out from the previous set; the final metrics is the average value of the k computed metrics.

Machine learning models performances are sensible to the samples of the validation set; small variations in the set can cause relevant changes in performance; with this techniques, the over-fitting problem is reduced because every data in the training set is applied to validate a model at least one time.

Such training strategy is also used to reduce the variance of the errors especially if the size of the dataset is small; the approach can look computationally expensive, but it does not waste too much data which is a major advantage for this problem ([33]).

Chapter 4

Interesting Experiments

The designing of the loss function guided our work into several experiments. In this chapter we collected their results; we acted to train a module which is able of predicting the coating weight from the context variables of the process line; we named such predictor *Coating Network*.

4.1 Coating Net designing

The model takes in input the values $< t, h, s, p, d >$ and estimates the corresponding coating value $< \hat{c} >$. At the beginning, we started our study trying to predict the coating by using a math formula of the physical phenomenon; for this, we conducted a mathematical study of the phenomenon which lead us to a linear regression model;the study is presented in the next section; then, we compared four different neural network architectures with such linear regression model and the standard predictor; we tested them for each coating group. The description of their structure, the way on which they were trained, and the results of the experiments are showed at the section Coating Net construction.

4.2 Phenomenological approach for wiping process control

We applied the same approach of the previous work [16]: we studied the variables of the wipe problem as an empirical derivation of the phenomenon and we put them in the following mathematical equation:

$$c = \frac{s * d * h}{p} * \exp(t) \tag{4.1}$$

Such formula, can be rewritten in a logarithmic scale as the linear combination of the variables:

$$log(c) = log(s) + log(d) + log(h) + t - log(p) \tag{4.2}$$

The linear regression model can have a direct map from 4.2;
Such relation is:

$$log(c) = w_1 log(s) + w_2 log(d) + w_3 log(h) + w_4 t - w_5 log(p) \tag{4.3}$$

To conclude the study, in 4.4 we lead back the weights of regression model as coefficient of the variables in the initial formula 4.1:

$$c = \frac{s^{w_1} * d^{w_2} * h^{w_3} * \exp(w_4 t)}{p^{w_5}} \tag{4.4}$$

4.3 Coating Net construction

The steps we did to define the final architecture of the coating net are presented in this section. In total we designed and trained five different predictors, we started by describing four, the last one is described in the following sections concerning the experiments. They differ in their architecture or way of being trained:

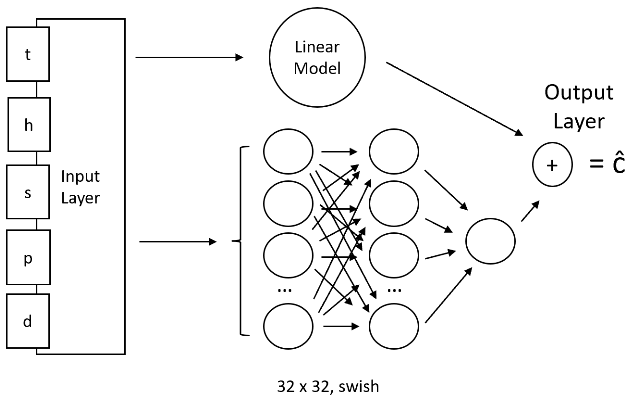


Figure 4.1: Coating Net Structure

- **Linear Regression Model (LR)**
  We designed a model according to the mathematical mapping that we proposed in the previous section: see formula 4.4; by using a single unit structure we trained a linear regression model; the model has no activation function and no bias; this two characteristics ensure a perfect matching with formula 4.3.

- **Two Dense Layers Regression Model (TDL)**
  The topology of this network is composed by a stack of 4 layers: input layer, 2 hidden dense layers of 32 units for each and the output layer on the top. The *swish* activation function for each layer is applied.

- **Coating Net V1 (CNV1)**
  The model consists of the combination of both previous models; its general structure has two towers that process the same input; in addition, the output layer of CNV1 receives the output tensor lists of both towers and returns a single tensor which is the sum of the results; we present the structure in the image 4.1. The structure of the first module of the network is the same of the LR model; the second tower, on the other hand, has the same topology as TDL; the only difference from the latter is that the output level of this module has no activation function.

MSE Losses				
Coat Group	LR	TDL	CNV1	**CNV2**
from 39 to 59	5.129	6.117	**4.883**	5.025
from 59 to 89	12.798	11.153	12.545	**9.635**
from 89 to 124	56.662	79.724	58.317	**28.590**
from 124 to 144	3.620	3.739	4.044	**3.298**

Table 4.1: MSE losses evaluation between LR, TDL,CNV1 and CNV2.

- **Coating Net V2 (CNV2)**
  This network has the same structure of CNV1 but, this time, we trained this model in a different way: one tower estimates the coating and the second one acts as reinforce block; it aims to add some correction value to the first tower; we refer to this second module as *"additive block"*. We trained first the LR model and then, we injected its weights in the linear tower and we froze it; finally, we trained alone the additive block from scratch.

4.4 Coating Net Evaluation

We conducted experiments for four different coating groups. This led us to split the dataset into four smaller set; as each set exhibited different behavior during training, we adapted the following parameters: initializer, regularizer, learning rate, batch size and epoch numbers, according to that behavior. We found that initializing the weights with *He-Normal* Initializer or *RandomUniform(minval=-0.5,maxval=0.5)*Initializer speeded up the training; in addition, to avoid overfitting problem we used kernel regularizer *L2* techniques and the *5-fold Cross Validation* method. The explanation of the technique is provided at section 3.9; the losses showed in the next section are taken as the average of the five models generated via the cv techniques.

Output distributions of the additive block						
	CNV1		CNV2		CNV3	
Coat Group	μ	σ	μ	σ	μ	σ
from 39 to 59	18.42	1.05	0.99	1.01	2.72	1.00
from 59 to 89	18.27	1.06	1.00	1.03	2.57	1.02
from 89 to 124	36.69	1.07	1.00	1.05	2.51	1.13
from 124 to 144	21.47	1.05	0.99	1.00	2.72	1.00

Table 4.2: Output additive block distribution

4.4.1 First Experiment

The architectures has been trained into four ranges of coating: the first experiment compares their *mean squared error (mse)* losses performed in the test-sets. The results are showed in the table 4.1; CNV2 outperforms all the models for each coating group; the second consideration which is worth to highlight is that LR and CNV1 look to have similar losses values. For this reason, we investigated about the sustainable difference in losses between CNV1 and CNV2.

4.4.2 Second Experiment

To understand better what the two models learnt during the training, we used μ and σ to describe the distribution of their additive block outputs; the results presented the table 4.2 are the output values obtained in grams of coating zinc. The CNV2 mean output lies around 1 grams for each group, on the other hands, CNV1 produce many grams more; such big difference between the two is the indicator that the two networks learnt a different behavior. However, to be sure that the training approach used by CNV2 is better then CNV1 we conducted a third experiment with a new version of *Coating Net*. In table 4.2 we also anticipated the results of the next experiments. The output of such new version lies near to the CNV2 values.

MSE Losses		
Coat Group	**CNV2**	CNV3
from 39 to 59	5.025	**4.705**
from 59 to 89	**9.635**	11.761
from 89 to 124	**28.590**	32.066
from 124 to 144	**3.298**	3.458

Table 4.3: MSE evaluation for model CNV2 and CNV3.

4.4.3 Third Experiment

We wanted to force CNV1's additive block to learn the same behavior as CNV2 from scratch; hence, we added to the output layer of its additive block the *tanh* activation function and then, we trained again from scratch the model; we called such version of **Coating Net** as **Version 3 (CNV3)**.

We compared the two architecture losses in the table 4.3; CNV2 still is doing better in loss evaluation; in addition we can also see in table 4.2 that the output of the additive block of CNV3 lies around two; the values of 2 can be misunderstood with tanh output property and needs an explanation: what the activation function clips are values at a logarithmic scale; the log of CNV3 values is 0.4 which results widely in the range [-1,1]; the CNV2 approach produced log values around 0.007; the CNV3 had the opportunity to learn the same behavior of CNV2 but it didn't worked as expected. With these new results, we can conclude that the learning approach showed in CNV2 works better among the five different predictors' architectures.

4.4.4 Fourth Experiment

Until now, the approach we followed for CNV2 concerned the training of a machine learning model for each of the different coating groups. However, four predictor models are difficult to maintain; furthermore, the dataset of a single coating group may be not enough to train a neural network properly; for this reason, we decided to train the CNV2 architecture for all the groups.

MSE Losses			
Coat Group	Coating Net	CNV2	Standard Model
from 39 to 59	6.640	**4.741**	4.941
from 59 to 89	15.436	**9.613**	14.255
from 89 to 124	**24.727**	29.891	75.0727
from 124to 144	28.165	**3.863**	4.149
All groups	17.525	**12.027**	24.604
Output additive block			
Coating Net		CNV2	
$\mu = 1.0085$	$\sigma = 1.0465$	$\mu = 0.9996$	$\sigma = 1.0238$

Table 4.4: MSE evaluation and output shape of the additive blocks.

From now on, since we chose it, we refer to this new predictor as *Coating Net*. In Table 4.4 we compared the *mse* losses of the two final models with the *Standard Model*. In this work we have not applied proprietary software except for comparisons. For the sake of curiosity, we have also added at the bottom of the table, the shape of the output of the additive block for both nets: *Coating Net* and CNV2.

From the results in the table we can see that grouping outputs into coating groups is a general technique that can pay off. Another consideration is that the CNV2 approach is the most accurate: it's general *mse* is lower in on "all groups" row.

The *Coating Net*, due to the generalization of groups, of course lost something in accuracy; however, we still consider it a valid approach because its average result in *mse* is lower than the *Standard model*.

Finally, CNV2 and *Coating Net* have a distribution of values with a very similar shape at the output of their *additive blocks*.

Chapter 5

Controller Nets: Training and Evaluations

Before to focus on this chapter we should briefly review some important concepts which lead our project scope. The current control system, which in use in the hot dip galvanizing process line, provides a strong accuracy in prediction; in chapter 4 we already compared it with the *Coating Net* model; anyway, for design reasons, its logic of control of the nozzles doesn't work properly sometimes: the requirements we evaluated in chapter are 3 not satisfied and the solutions it generates, are sub-optimal; this brought us to our *Controller*.

In this chapter we are going to describe the influence of the weights as loss components; how we planned the training phase; the metrics we used to validate the model; and finally, we reported the evaluations tests in precision and optimization in the last paragraphs.

5.1 Training parameters

This section reports the paramenters we applied to train the *Controller Network*. Initialization helped to speed-up the training phases of the networks; we instantiated the *Random Uniform* Initializer class of Keras library:

min-value 0.045, max-value 0.065 and seed=42. Furthermore, to avoid over-fitting problems we applied in combination the *L2* regularization penalty of 0.0001 on the layer's kernel and *Early Stopping Callback* methods.

The *Controller* was trained with a learning rate of 0.001 and batch size of 64. They worked good for each strategies.

5.2 Validation Metrics

During the training of the model, we have tried to keep the accuracy of the model close to that of the current predictor. Furthermore, we have tried to provide a new logic for piloting the *airknives effect* towards solutions $< \hat{p}, \hat{d} >$ that are able to produce better quality final products. Three outlooks are taken into account by the validation metrics we engineered:

1. The coating loss produced by the tuples. We prioritized it w.r.t. the other components of the loss. We focused on reducing it, of course.

2. We acted to maximize the decrement in pressure and distance values from the current solutions. It means increasing the optimization.

3. We compared the accuracy of the controller with the standard model outputs when applied in some few optimal cases.

5.3 Training strategies

Each component in the total loss adds some features to the final learned behaviour; the influence of such characteristics in the function is defined directly by the size of its weights: high weights can provide strong influence in the loss computation; lower weights can results too week (or small) to be considered in the sum of the total loss.

Furthermore when one features has too much weight compared to the others, its influence can mask some behaviors; as a consequence, at the end of the training such behaviors are not learned.

We planned to compare three different strategies of training; each strategy leads to a different model; in the following sections we will described them and how we tuned the model for each of them. The general approach, which we followed during the tuning of the weights w_1 and w_2 of the loss (formula 3.1, chapter 3), was trying small values first; this ensured that the coating target was prioritized; then, we increased the weights value exponentially until we had a balancing in all validation metrics.

First Strategy: Minimize the pressure

The first approach was to model only the minimization of the pressure values; we set $w_2 = 0.0$ and in this way, we reduced to 0 also the minimization component of the distance;hence, we tried different values of w_1. At the end, it worked fine for $w_1 = 0.3$.

Second Strategy: Minimize the distance

This second approach was to model only the distance values; hence, we let the network appropriately reduce the pressure in an indirect way; we tuned $w_1 = 0.0$ to have the minimization component of the pressure to 0. From our trial and error approach, it was found that this strategy worked well for $w_2 = 10.5$. The weights values between the two strategy are very different because pressure and distance components work in a different scale of values.

Third Strategy: Double weights tuning

In this approach we tuned both weights as $w1 = 0.125$ and $w2 = 5.25$. To discovered them we divided by two the weights we found before, and then, we did some trials to tune accordingly.

5.4 What Controller Net learned

In this section our aim is to provide to the reader a full comprehension of what the *controller net* can learn by applying one of the strategies that we proposed above. Specifically in the image 5.1, we showed the output of the one which was trained with *Strategy 3*; the other two strategies are very similar; we didn't evaluated big differences which are worth to be showed; the evaluation of the differences of three strategies is presented in the next sections; furthermore, anticipating the results, *Strategy 3* will be the favorite approach; for all this reasons, we decided to show only this images.

Before to continue, a brew explanation of how what is showed in the image is needed: we fixed first the value of height to 650 mm and the Temperature around 450°C; subsequently, we fed into the *Controller Net* increasing values of speed and different coatings; the values we used as coating target are the mean values of each group that was presented in the previous chapters.

In the x-axis there is the speed range which goes from 50 m/s to 140 m/s; in the y-axis, the pressure and distance normalized values into the range [0,1]; we did the conversion to bring the values into the same unit scale. The dashed lines in the Cartesian plane are the output of the distance; the solid ones are the output of the pressure; the lines: dashed and solid, with the same color has the same target coating. The vertical black line is the speed threshold; it is a reference for indicating when the minimum pressure changes.

According to the physics of the problem: if the speed increase, to maintain the same value of coating the *Controller* should increase the pressure or decrease the distance. In the image we can see that this behavior is proportionally executed: near to the origin we have low values of speed; in this case the net keeps the value of the pressure at the minimum level and conversely, the value of distance is high in the range of its values; it is actually around 13/14 mm. As speed increases, the network to maintain the same coating reduces the distance; When the distance arrives to the minimum possible value, the only thing that the network can do is increasing the pressure; and

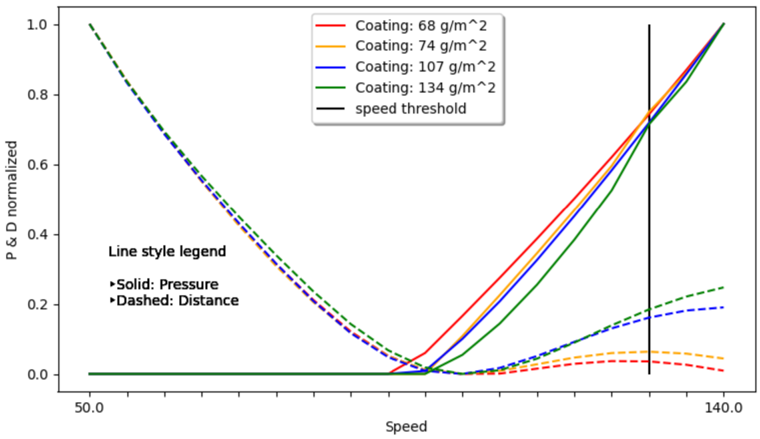


Figure 5.1: Controller Net: The output behaviour

it is done too.

In the image there are some events which occurred: for the low coating value (red color) the *controller* increases the pressure before of high coating values (green); this behavior is reasonably and physically interpretable; furthermore, in the case of the red coating, the pressure starts to increase before the distance is completely arrived at the minimum. Finally, we can see that the network increases pressure and distance by weighing them proportionally. The order of the coating is respected in both pressure and distance outputs. When the threshold of the speed is passed though, there is a change in the reference of the minimum values of pressure. The *controller* maintained the order of the coating values even for higher speeds.

5.5 Experimental Evaluation

We tested the models over the unseen data of the dataset. We measured both precision and optimization efficiency. In the next two sections we provided the explanations of the test results.

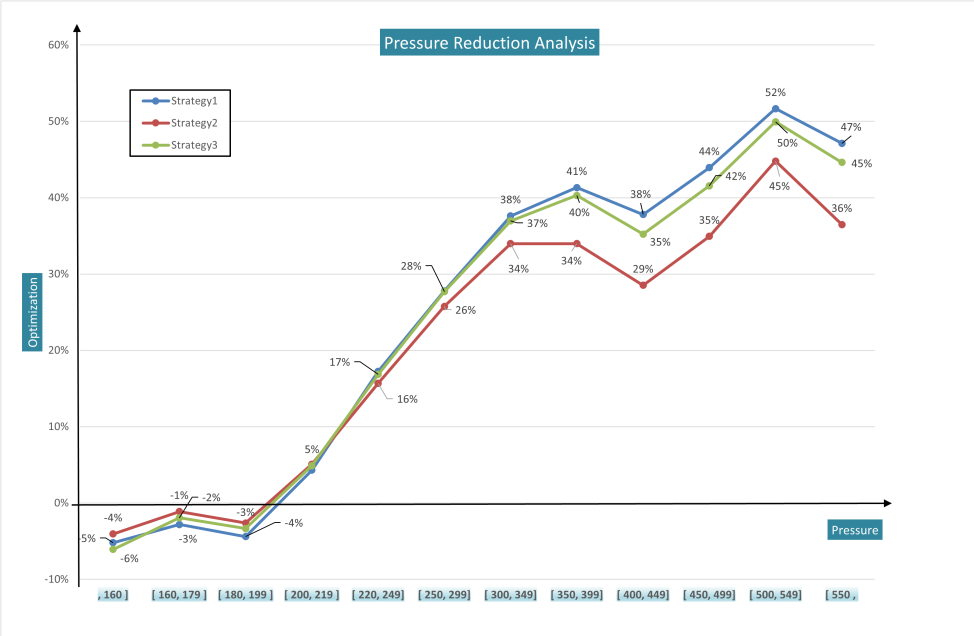


Figure 5.2: Percentage reduction of pressure values

5.5.1 Optimization test

We conducted a study on the efficiency of the three optimization policies we gained from the learning strategies: results are showed in the figures 5.2 and 5.3. The images confirms the effectiveness of our three strategies approach. On the y-axis of both graph we put the degree of optimization expressed as percentage of decrement for both values: pressure (in fig. 5.2) and distance (in fig. 5.3); on the other hands, on x-axis, we binned the values of distance or pressure applied by the current solutions logic.

According to the results in the image 5.2, when the operator works around 520 millibars of pressure, the controller Strategy 1, produces an alternative working tuple $< p, d >$ with a pressure, for instance, *52%* lower; i.e.: the controller suggests of applying 270 millibars of pressure instead of 520, and gives a corresponding distance to obtain the same target.

From the image, we can also conclude that the leader in pressure optimization, as we expected, is the controller trained with Strategy 1.

Figure 5.3: Percentage reduction of distance values

Differently, if we look into the image 5.3, Strategy 2 leads the distance reduction analysis.

The bigger reduction in distance values, is provided by Strategy 2 when the operators worked around 11 millimeters; the new distance which is proposed, in this case, is 2.3 millimeters lower than the current value.

The results of both graphs paved a promising path: the trend of the reductions is crescent. Furthermore, the negative percentage at the beginning of both graphics, and the hole, which is almost in the middle of the rising part of the lines, are due to the two safety requirements: (1)The network was designed not to exceed the minimum distance of 7.5 mm and,(2) the 160 or 250 millibar of pressure according to the speed of input.

In order to accept such optimization values, however, it is important to perform also accurate precision tests; they make us understand which network works better in accuracy; such tests are presented the next two sections.

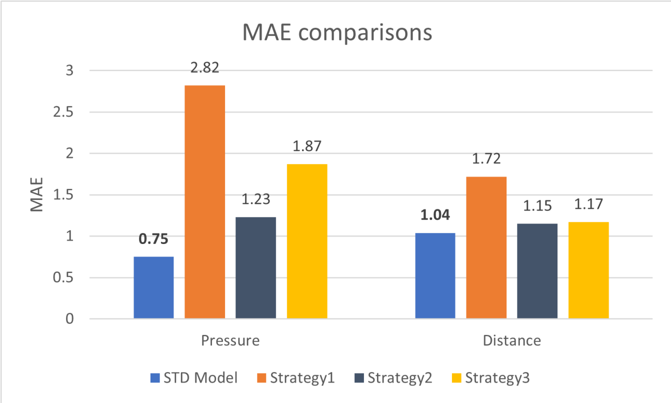


Figure 5.4: Mae comparisons

5.5.2 Accuracy tests on real optimal use cases

We pulled out from historical sets of Marcegalgia, the cases where operators worked at minimum conditions in pressure and distance. Such cases have a speed always lower than 109 m/m (meters of strip per minute) and a pressure lower than 170 millibar; naturally, they are not enough to cover all the space of the optimal solutions, however, they are valuable cases which is worth to test.

At this point we had the opportunity of comparing one-to-one the output of the controllers with the *Standard Model* and, as a consequence, we had also the opportunity for testing the accuracy of the predictions for both outputs $\hat{p}$ and $\hat{d}$.

The picture 5.4 shows the *mae* (mean absolute error) values that we get from the test; the *standard model* won in both error values: pressure and distance; anyway, what counts here is the level of general accuracy of the networks; if we focus on this, we see similar losses among the bars in the chart; the worst result of *mae* is for the pressure of Strategy 1; It differs of just less than 2.5 millibars from the *mae* of the *standard model.* Such small differences in the chaotic context of the production line can be of course imperceptible.

Confirms in this direction arrived also from another point of view of the same test: we evaluated the accuracy by counting the number of time the networks produced satisfactory pressures and distances; according to domain experts, we defined as "draw between two solutions" when the difference in pressure between the output of two models does not exceed a tolerances of 6 millibars of pressure.

We got that networks provided bad outputs only in a very few cases:

- *Strategy 2*: The network drew or did better every time: **100%**.
- *Strategy 3*: The controller drew or did better the **98.33%** of the time; it lost few times.
- *Strategy 1*: The model tied with the standard model **90.0%** of the times (,but never won against). However, by relaxing both tolerance parameters: distance=0.3 and pressure=7; **94.17%** of ties are obtained.

From such results it is possible to conclude that all the three controllers in the test, returned values which are not distant from the standard model. With some values of tolerance more, even Strategy 1 can achieve a valuable level of reliability and precision.

5.5.3 Accuracy tests on new generated solutions

When the controller generates a tuple of values which was never met in production, performing the test as we did before is not possible; thus, we evaluated the new generated solutions by using two tools: the *Coating net* and the *Standard model*; we presented their *mse* losses in the table 4.4 of chapter 4.

The estimated metrics for this test is the mean differences of zinc grams at each group coating target; such metric can be seen also from another point of view: the general coating loss for each group. The results are showed in the pictures 5.6 and 5.5.

In the bar graphs, negative values refer to the grams missing to reach the coating target; positive values are excess grams; the domain expert, even

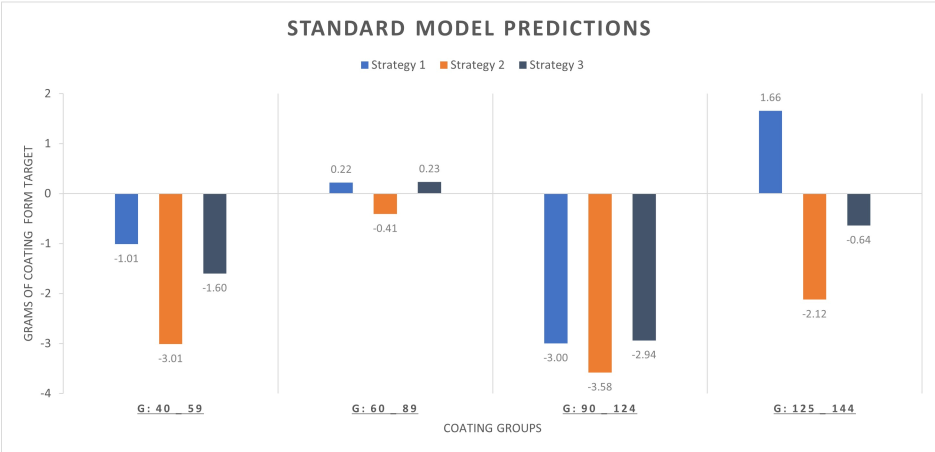


Figure 5.5: Standard model: difference in grams between the expected and the target coating.

in this case, defined the value of ± 3.0 grams as the acceptable level of prediction; this value was also mentioned in chapter three in the requirements analysis.

According to the prediction of the *standard model*, such bound is not respected for few tenths of grams by Strategy 2 only in one case: when the model targets the coating group *G: 90_124*; however, Strategy 2 is still valuable because the test in table 4.4 showed the highest *mse* loss in the prediction of the tool for such group.

However, from the image 5.5 we see that the prediction made by the *standard model* test are favorable to Strategy 3. It did better in the two groups with the largest coating. Furthermore it performed very well also for the target group *G: 60_89*.

Finally, to conclude this section, in 5.6 we showed the second evaluation which this time was made by applying the *Coating Net* model; the test result is in favour of Strategy 3 for three of the four groups.

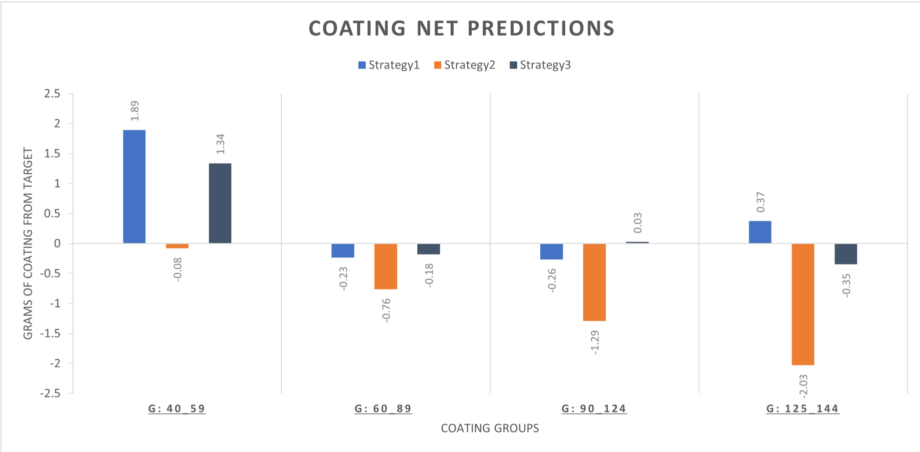


Figure 5.6: Coating net forecast: difference in grams between the expected and the target coating.

5.5.4 Final considerations on results

The standard model is very accurate in the output of pressure and distance; however, it has a current logic of minimum which is not largely comparable with the optimization we proposed in this work; among the three training strategies we proposed for controllers, we saw in the results of fig. 5.2 and fig. 5.3 that Strategy 3 is able to capture the optimization of both output parameters: pressure and distance.

Secondly, we carried two accuracy tests which showed promising results: Strategy 2 and 3 produced very similar outputs to the standard model for real optimal solution cases (100% and 98.33%).

Finally, we estimated also the projections of the new values that the logic proposed via the coating net and standard model. From such final test, Strategy 3 is the controller which achieved the minimum coating differences for the largest number of groups in both prediction tools.

However it is also important to consider that the three proposed logics can be adapted to the purpose of optimization; we saw that all the three models worked with good level of precision and the estimation of the target coating they receive in input is in the rage of $\pm$ 3 g/m^2;

Chapter 6

Conclusions and Future works

In this thesis we presented a new approach to solve a real problem in the intelligent steel industry: to optimize the autonomous control of the weight coating modeling applied by the air-knife effect in a hot dip galvanizing line.

A formalization of the problem was provided; requirements get collected; furthermore two important analysis were carried out: the measurements taken during the steps of the process line; and the lacks of the current control systems.

The result was a deep learning model which we called controller; and the techniques we used to train it; the controller is able to drive the air-knife effect towards optimized solutions that lead to the coating weights provided in its input.

The loss function we envisioned concerns the minimization of three components: (1) the error on the expected resulting coating; (2) the distance of the pressure from its minimum values;(3) and the minimization of the distance from its minimum values.

The loss required also the implementation of a second neural network to provide the ground truth on which compute the error with the expected resulting coating; we called such neural network coating net.

During its construction we compared four different models with the standard model; the most accurate was the architecture concerning a structure made by a linear and a deep component. The linear reflects an existing mathematical approach and makes prediction of the output; in our cases, the deep nonlinear "residual" component learned from empirical observations to adjust such output.

The advantage of the presented loss function is that by tuning the weights of the different components of the loss function, it is possible to train model with slightly different optimization purposes.

Tests showed mainly two things: the first is that the logic provided by the controllers can optimize of 50% the sub-optimal values of pressure and of the 20% some distance values; the second thing is that all the three strategies can be regularized with a good level of precision. The accuracy of the coating target is in the range of $\pm$ 3 g/m^2;

6.1 Future works and open challenges

The future extension of the work of this master thesis concerns mainly three outlooks:

The first one is that the Coating net architecture lost something in precision during its construction; we have to go in this direction and create a much robust controller net; however, our favorite solution we device to face such problem is of *implementing the two models: Controller and Coating net in continuous learning;* the event-drive context of the production line is opportune to such kind of application.

Secondly, a possible future work is about *feeding the model with new information that can have important impact on weight zinc control;* they could be for instance: outdoor Temperature and Humidity; a chemical representation of the zinc; The presence of sodium in the zinc; etc.

Finally, the latest possible direction could be to extend the loss by considering the moving of the formalism we made simplified maybe too much the real problem; the open challenge is to solve the same problem but taking into account two additional considerations: the steel strip can move from the perfect center of the two pumps and; the nozzles movement in distance and height is not symmetric anymore; They can move arbitrary.

Bibliography

[1] Daniel Alvarez, Ana Cerezo-Hernández, Graciela López-Muñiz, Tania Castro, Tomas Albi, Roberto Hornero, and Felix del Campo. *Usefulness of Artificial Neural Networks in the Diagnosis and Treatment of Sleep Apnea-Hypopnea Syndrome*, pages 33–68. 04 2017.

[2] K.J. Åström and R.M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2010.

[3] Mariette Awad and Rahul Khanna. *Support Vector Machines for Classification*, pages 39–66. 01 2015.

[4] D.J. Bartholomew. Analysis and interpretation of multivariate data. In Penelope Peterson, Eva Baker, and Barry McGaw, editors, *International Encyclopedia of Education (Third Edition)*, pages 12–17. Elsevier, Oxford, third edition edition, 2010.

[5] Yoshua Bengio. *Deep Learning*. Adaptive Computation and Machine Learning series. MIT Press.

[6] Massimo Bertolini, Davide Mezzogori, Mattia Neroni, and Francesco Zammori. Machine learning for industrial applications: A comprehensive literature review. *Expert Systems with Applications*, 175:114820, August 2021.

[7] Gundula Bosch. Train PhD students to be thinkers not just specialists. *Nature*, 554(7692):277–277, February 2018.

[8] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[9] Jose Calaf-Chica, P. Bravo, and Mónica Preciado. A new prediction method for the ultimate tensile strength of steel alloys with small punch test. *Materials*, 11:1491, 08 2018.

[10] Antonio Ciampi. Generalized regression trees. *Computational Statistics Data Analysis*, 12(1):57–78, 1991.

[11] Michele Compare, Piero Baraldi, and Enrico Zio. Challenges to iot-enabled predictive maintenance for industry 4.0. *IEEE Internet of Things Journal*, 7(5):4585–4597, 2020.

[12] F. J. Martínez de Pisón, A. Pernía, E. Jiménez-Macías, and R. Fernández. Overall model of the dynamic behaviour of the steel strip in an annealing heating furnace on a hot-dip galvanizing line. *Revista de Metalurgia*, 46(5):405–420, October 2010.

[13] R. E. Elewa, S.A. Afolalu, and O.S.I Fayomi. Protective impact of molten zinc coating sheets in contaminated environment-review. *Journal of Physics: Conference Series*, 1378(2):022071, December 2019.

[14] E.A. Elsaadawy, G.S. Hanumanth, A.K.S. Balthazaar, J.R. McDermid, A.N. Hrymak, and J.F. Forbes. Coating weight model for the continuous hot-dip galvanizing process. *Metallurgical and Materials Transactions B*, 38(3):413–424, May 2007.

[15] Ahmed Farahat, Ahmad Gemeal, and Ramadan Elshaer. Influence of skin pass design and aging phenomena on steel sheets surface characteristics. *Journal of Failure Analysis and Prevention*, 16, 12 2015.

[16] A. Fiorini, A. Ferraiuolo, A. Ollagnier, M. Monnoyer, P. Rocabois, S. Pantarotto, and E. Menigault. Automatic pilot for strip processing lines. In *AISTech2020 Proceedings of the Iron and Steel Technology Conference*. AIST, 2020.

[17] Christian Fuchs. Industry 4.0: The digital german ideology. *TripleC*, 16:280–289, 01 2018.

[18] Zahra Ghasemivinche and Ali Zeinal Hamadani. Predicting mechanical properties of galvanized steels: Data mining approach. *International Journal of Advanced Engineering, Management and Science*, 3(7), 7 2017.

[19] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[20] N. Guelton and A. Lerouge. Coating weight control on arcelormittal's galvanizing line at florange works. *Control Engineering Practice*, 18(10):1220–1229, 2010.

[21] Nicolas Guelton, Catherine Lopès, and Henri Sordini. Cross coating weight control by electromagnetic strip stabilization at the continuous galvanizing line of ArcelorMittal florange. *Metallurgical and Materials Transactions B*, 47(4):2666–2680, April 2016.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[23] Khyati Patel Sweta Patel Jayraj Parmar, Daulat Kumar Sharma. A review on galvanizing coating defects: Causes and remedies. *Jurnal Kejuruteraan*, 34, 2022.

[24] Gennady P. Kornilov, Ildar R. Abdulveleev, Timur R. Khramshin, and Valery V. Shokhin. Advanced electric drive control system of continuous hot-dip galvanizing line. In *2020 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*, pages 1–6, 2020.

[25] Arun K. Kuchibhotla, Lawrence D. Brown, Andreas Buja, and Junhui Cai. All of linear regression, 2019.

[26] Yong-Hun Kweon and Heuy-Dong Kim. Study on the wiping gas jet in continuous galvanizing line. *Journal of Thermal Science*, 20(3):242–247, May 2011.

[27] Ivano Lauriola, Alberto Lavelli, and Fabio Aiolli. An introduction to deep learning in natural language processing: Models, techniques, and tools. *Neurocomputing*, 470:443–456, 2022.

[28] Yuan-Pin Lin and Tzyy-Ping Jung. Improving EEG-based emotion classification using conditional transfer learning. *Frontiers in Human Neuroscience*, 11, June 2017.

[29] Zhihao Liu, Quan Liu, Wenjun Xu, Lihui Wang, and Zude Zhou. Robot learning towards smart robotic manufacturing: A review. *Robotics and Computer-Integrated Manufacturing*, 77:102360, 2022.

[30] Liangqu Long and Xiangming Zeng. *Keras Advanced API*, pages 283–314. Apress, Berkeley, CA, 2022.

[31] A.A.O Magalhães, I.C.P Margarit, and O.R Mattos. Electrochemical characterization of chromate coatings on galvanized steel. *Electrochimica Acta*, 44(24):4281–4287, 1999.

[32] Kai Mao, Yong-Li Yang, Zhe Huang, and Dan-yang Yang. Coating thickness modeling and prediction for hot-dip galvanized steel strip based on ga-bp neural network. In *2020 Chinese Control And Decision Conference (CCDC)*, pages 3484–3489, 2020.

[33] Natalie R Masse. A review of statistical learning methods with applications. 2022.

[34] Yujian Mo, Yan Wu, Xinneng Yang, Feilin Liu, and Yujun Liao. Review the state-of-the-art technologies of semantic segmentation based on deep learning. *Neurocomputing*, 493:626–646, 2022.

[35] Paul Nurse. Biology must generate ideas as well as data. *Nature*, 597(7876):305–305, September 2021.

[36] F. Ozturk, Z. Evis, and S. Kilic. 3.12 hot-dip galvanizing process. In MSJ Hashmi, editor, *Comprehensive Materials Finishing*, pages 178–190. Elsevier, Oxford, 2017.

[37] Deepankar Pal, Amlan Datta, and Satyam S. Sahay. An efficient model for batch annealing using a neural network. *Materials and Manufacturing Processes*, 21(5):567–572, August 2006.

[38] Zai-Sheng Pan, Tian-Lin Wang, Shao-Liang Wang, and Peng Chen. Neural network based modelling and prediction for the coating weight of hdgl process. In *2017 36th Chinese Control Conference (CCC)*, pages 2122–2126, 2017.

[39] Thijs Peirelinck, Hussain Kazmi, Brida V. Mbuwir, Chris Hermans, Fred Spiessens, Johan Suykens, and Geert Deconinck. Transfer learning in demand response: A review of algorithms for data-efficient modelling and control. *Energy and AI*, 7:100126, 2022.

[40] Edgar O. Reséndiz-Flores, Gerardo Altamirano-Guerrero, Patricia S. Costa, Antonio E. Salas-Reyes, Armando Salinas-Rodríguez, and Frank Goodwin. Optimal design of hot-dip galvanized dp steels via artificial neural networks and multi-objective genetic optimization. *Metals*, 11(4), 2021.

[41] Daniel A. Roberts, Sho Yaida, and Boris Hanin. *The Principles of Deep Learning Theory.* Cambridge University Press, 2022. `https://deeplearningtheory.com`.

[42] A. Sanz-García, J. Fernández-Ceniceros, R. Fernández-Martínez, and F. J. Martínez de Pisón. Methodology based on genetic optimisation to develop overall parsimony models for predicting temperature settings on annealing furnace. *Ironmaking &amp Steelmaking*, 41(2):87–98, December 2013.

[43] Martin Saxinger, Lukas Marko, Andreas Steinboeck, and Andreas Kugi. Feedforward control of the transverse strip profile in hot-dip galvanizing lines. *Journal of Process Control*, 92:35–49, 2020.

[44] Shashi Shekhar, Hui Xiong, and Xun Zhou, editors. *Exploratory Data Analysis*, pages 570–570. Springer International Publishing, Cham, 2017.

[45] Zai sheng Pan, Xuan hao Zhou, and Peng Chen. Development and application of a neural network based coating weight control system for a hot-dip galvanizing line. *Frontiers of Information Technology &amp Electronic Engineering*, 19(7):834–846, July 2018.

[46] Ki-Tae Shin, Hea-Doo Park, and Wan-Kyun Chung. Synthesis method for the modelling and robust control of coating weight at galvanizing process. *ISIJ International*, 46(10):1442–1451, 2006.

[47] A. Yahyaee Soufiani, J. R. McDermid, A. N. Hrymak, and F. E. Goodwin. Experimental and numerical study of coating thickness using multi-slot air knives. *Metallurgical and Materials Transactions B*, 50(6):2523–2535, August 2019.

[48] Onawale Tairu, P. Aiyedun, and O.T. Tairu. Relationship between yield stress and yield strength on various grade of steel being hot rolled. *IOSR Journal of Mechanical and Civil Engineering*, 11:40–46, 01 2014.

[49] John A. Thornton and Hart F. Graff. An analytical description of the jet finishing process for hot-dip metallic coatings on strip. *Metallurgical Transactions B*, 7(4):607–618, December 1976.

[50] Murat Tiryakioḡlu. On estimating the fracture stress and elongation of a1-7alloy castings with single pores. *Materials Science and Engineering A*, A527:4546–4549, 07 2010.

[51] Nilesh Tripuraneni, Michael Jordan, and Chi Jin. On the theory of transfer learning: The importance of task diversity. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7852–7862. Curran Associates, Inc., 2020.

[52] C.V. Tu and D.H. Wood. Wall pressure and shear stress measurements beneath an impinging jet. *Experimental Thermal and Fluid Science*, 13(4):364–373, November 1996.

[53] Rolf Werner, Dimitar Valev, and Dimitar Danov. The pearson's correlation -a measure for the linear relationships between time series? 09 2009.

[54] Darrell Whitley and Andrew M. Sutton. *Genetic Algorithms — A Survey of Models and Methods*, pages 637–671. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[55] Yuanlun Xie, Majun He, Tingsong Ma, and Wenhong Tian. Optimal distributed parallel algorithms for deep learning framework tensorflow. *Applied Intelligence*, 52(4):3880–3900, July 2021.

[56] Shangshang Yang, Ye Tian, Cheng He, Xingyi Zhang, Kay Chen Tan, and Yaochu Jin. A gradient-guided evolutionary approach to training deep neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33(9):4861–4875, 2022.

[57] Zhen Yu, Jiming Hu, and Huimin Meng. A review of recent developments in coating systems for hot-dip galvanized steel. *Frontiers in Materials*, 7, 2020.

[58] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. Dive into deep learning, 2021.