Recurrent Neural Network Based Approach for Estimating the Dynamic Evolution of Grinding Process Variables

Ander Arriandiaga Laresgoiti

PhD Thesis – 2016

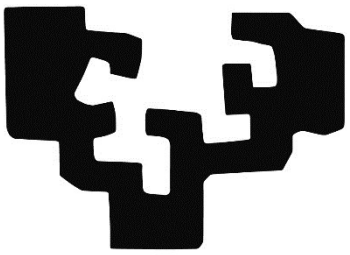
eman ta zabal zazu

Universidad del País Vasco

Euskal Herriko Unibertsitatea

RECURRENT NEURAL NETWORK BASED APPROACH FOR ESTIMATING THE DYNAMIC EVOLUTION OF GRINDING PROCESS VARIABLES

eman ta zabal zazu



Universidad del País Vasco

Euskal Herriko Unibertsitatea

Ander Arriandiaga Laresgoiti

Department of Automatic Control and System Engineering

University of the Basque Country – Euskal Herriko Unibertsitatea

Supervised by

Eva Portillo and José Antonio Sánchez

This dissertation is submitted for the degree of Doctor of Philosophy

October 2016

Hemen aurkezten dudana ez da nik bakarrik egindako lan bat, beste askoren laguntzarekin hasitako ibilbide baten bukaera baizik.

Lehenik eta behin, nire zuzendariak izan diren Eva Portillo eta José Antonio Sáncez-i eskerrak eman nahiko nizkieke. Inoiz ez zitzaidan burutik pasatu noizbait Doktoradutza bat egin nuenik. Hala ere, beraiek nigan konfiantza eduki zuten eta lan hau bururaino eramaten lagundu didate. Beraz, beraien laguntza eta konfiantzarik gabe ezinezkoa litzateke hemen daukazuen lan hau aurkeztea. Naiz eta nire zuzendaria ez izan, eskerrak eman nahiko nizkioke Itziar Cabaneseri emandako laguntzarengatik eta hasitako bide honen parte garrantzitsua izan delako.

Era berean, eskerrak eman Sistemen Ingeniaritza eta Automatika Saileko kideei urte guzti hauetan eman didaten laguntzarengatik eta, nola ez, Ingeniaritza Mekanikoa Sailekoei, batez ere, tailerrean buru belarri lanean lan hau burutzeko beharrezkoak ziren esperimentuak batzen aritu direnei.

Ezin ditzaket ahaztu ama, aita, arreba eta Ladis etxean emandako animoengatik. Nolatan ez, eskerrak eman Equipo Actimel, koadrila eta Ieko eta ez-Ieko (Txaporta) lagunei.

Finally, I would like to thank Professor Kasabov, Joyce and all the staff from KEDRI for making me feel like one more. I cannot forget my friends of New Zealand. Thanks to all of you for making my stay in Auckland an unforgettable experience.

Guztioi, bihotzez, eskerrik asko!

"Imagination is more important than knowledge. For knowledge is limited to all we now know and understand, while imagination embraces the entire world, and all there ever will be to know and understand."

Albert Einstein

CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

Abrasive material removal processes are nowadays key technologies in modern manufacturing. Amongst them, grinding has become critical despite of the advances in performance and accuracy of other manufacturing processes, such as turning or milling. It is due to its capacity for producing parts of high precision and high surface quality in difficult-to-machine materials that grinding is widely used in motor industry, aerospace and precision cutting tool manufacturers (Malkin & Guo, 2008), amongst others. The application in those high-added value sectors easily explains why grinding has been the object of extensive research during the past 40 years and it is still nowadays.

Ground components are characterized by their extremely tight dimensional tolerances and very low surface finish. Grinding is, therefore, a high-accuracy technology in which process set-up, control and monitoring are of primary importance if the strict customer requirements are to be economically met.

Setting-up of the grinding process involves time and cost-consuming operations. Theoretical models have been developed, but industrial application is not an easy task yet. Theoretical models require calibration before they can be used in practical grinding operations. Examples of such models can be found in (Marinescu, et al., 2006). All these facts have led many researchers to use intelligent techniques in order to model the grinding process.

However, almost all the research works that use intelligent techniques found in the scientific literature provide particular solutions for a given wheel-workpiece pair. Actually, results cannot be generalized, in no case, to other types of grinding wheels not used during the design of the models. This is one of the reasons why industrial application of intelligent techniques in grinding has been very limited so far. Besides, in most cases, the output of the model is the value related to the current state of wear of the grinding wheel. However, the wheel loses its cutting ability over time.

Therefore, the objective of this work is to model the specific grinding energy, wheel wear and surface roughness with artificial neural networks due to their characteristics that made them suitable for modelling highly no-linear systems like grinding process. But unlike the works found in scientific literature, in this work the wheel wear, the surface roughness and the specific grinding energy are considered as dynamic evolutions with an initial point

(the wheel is sharp) and final point (the wheel is dull) i.e. taking into account the current state of wear of the grinding wheel. Besides, it would be necessary to generalize to new grinding wheels and grinding conditions in order to break the wheel-workpiece pair and develop models suitable for industrial applications.

Likewise, it is not the objective of this work to develop new artificial neural network architectures or training algorithms. Moreover, one of the objectives of this work is to use well-known artificial neural network architectures and training algorithms available in commercial software and analyse their limitations for modelling the complex relationship between the grinding wheel characteristics and operating conditions, and the grinding variables such as wheel wear, surface roughness or specific grinding energy.

The layout of the work is as follows. In Section 2 brief introduction of the grinding process is addressed in order to describe the basic aspects. Likewise, a description about different approaches found in literature for modelling the grinding process using analytical models and intelligent techniques is presented. Section 3 presents the artificial neural networks foundations. Subsequently, a review of the state of the art related to the dynamic evolutions modelling with artificial neural networks is made. The Section 4 is the main section of the work presented here. In this section the general strategy for estimating grinding dynamic evolution variables is presented. Thus, first, the artificial neural network configuration is presented. Then, the training strategy in order to find the best possible network is described.

Following the general strategy described in Section 4, in Section 5 smart sensors are developed for estimating on-line the wheel wear and the surface roughness. Besides the general description of the smart sensors, the achieved results are shown. Likewise, in Section 6, the prediction of the dynamic evolution of the specific grinding energy using the general strategy is described. As in the previous case, the results are showed.

After the conclusion addressed in Section 5 and Section 6, the first steps of a new line of investigation are described in order to overcome the limitations showed in previous Sections. Finally, in Section 8 the conclusions are drawn and in Section 9 the main highlighted lines for future investigation are described.

2 GRINDING PROCESS MODELLING

Grinding process has become key in manufacturing due to its capacity for producing parts of high precision and high surface quality in difficult-to-machine materials. In the grinding process the cutting tool is the grinding wheel. With the use, the grinding wheels wear and lose the cutting ability. Industrial application of the analytical models is not an easy task yet because of the lack of information about the composition of the wheels due to the semi-handmade production of them. Using intelligent techniques for estimation and/or prediction of outputs of the grinding process is a well-known approach. However, almost all the research works that use intelligent techniques found in the scientific literature provide particular solutions for a given wheel-workpiece pair. Besides, results cannot be generalized, in no case, to other types of grinding wheels not used during the design of the models. Finally, the output of the model is the value related to the current state of wear of the grinding wheel.

2.1 Grinding process

Machining tool processes are nowadays key technologies in a competitive global manufacturing marketplace (Kline, 2015). Although in 2015 the forecasting world machine tool consumption would fall slightly, it would still supposed to be of $75.0 billion (Kline, 2015). Thus, these can help to understand the importance of the machining tool processes.

Despite of the advances in the performance and accuracy of other machining tool processes for manufacturing, such as turning or milling, grinding process has become key in manufacturing. Actually, due to its capacity for producing parts of high precision and high surface quality in difficult-to-machine materials, grinding is widely used in high-added value sectors such as motor, aerospace industries or precision cutting tool manufacturing (Malkin & Guo, 2008).

For example, in Figure 1, a grinding machine for precision grinding of engine turbine blades is shown. In these extreme cases, where the clearance between the rotor blade tips

and the outer housing is critical on the performance of the engines, these machines held diametric tolerances of 0.025mm (DANOBAT, s.f.).

Figure 1 DANOBAT mBTG and DANTIP aerospace grinding machine series (http://www.danobatgroup.com/)

In the case of automotive industry, grinding process is used to produce steering components (wheel hub, tulip, tripod…), engine components (camshaft, crankshaft…) or gearbox (Doimak, s.f.) (Figure 2). For example, in China, the biggest machining tool producer and consumer (Kline, 2015), according to the statistics, more than 38% of the grinding process applications are for the automotive industry (Mao, 2014).

A)

B)

Figure 2 Automotive parts ground A) engine components B) Gearbox (Doimak, s.f.)

On the other hand, although grinding has been usually considered as a finishing process, it is also useful in some application fields as a material removal process. Actually, in aerospace, parts such as turbine blades (Figure 3), fuel injector needles or turbo charger shafts are examples that go directly from an investment casting to a finished product only using grinding (Tunstall, 2009).

Figure 3 Turbine blades machined with the grinding process (http://www.haas-schleifmaschinen.de/)

In the grinding process the cutting tool is the grinding wheel. It is composed by tiny abrasive grits bonded by some agent. The grinding wheel rotates at a constant speed and, thus, spin material is removed when it comes in contact with the part. The production of the grinding wheels is in a semi-handmade process. Therefore, it is very difficult to know *a priori* how the wheel will perform exactly during the grinding.

Figure 4 Abrasive grinding Wheel (http://www.abtec4abrasives.com/)

Grinding wheels main characteristics are (Rowe, 2009):

- Abrasive: The material used for generating the tiny particles to remove the spin material.
- Grit size: It makes reference to the size of individual abrasive grains. Thus, big grit size wheels are used for aggressive stock removal, while small grit size is used for less removal but better surface quality.
- Hardness: A bond material is used to hold together the abrasive grains. Thus, a strong bond is preferred for grinding softer material, while weak bond is preferred for harder material (Georgia Grinding Wheel Company, Inc., 2008). The grinding wheel hardness is rated from 'A' to 'Z', being 'A' the weakest bond and 'Z' the strongest.
- Structure: The structure refers to the density of abrasive grains in the grinding wheel. Wheel with less dense structure allows better swarf removal and gives better grinding fluid access (Rowe, 2009).
- Bond type: Is the material used to hold the abrasive grains. Bond type affects the finish. For example, Vitrified bond is suitable for precision grinding while resin

bonded wheels are used for rough grinding applications (Kure Grinding Wheel, 2016).

With the use, the grinding wheels wear and lose the cutting ability. In fact, the wheel surface becomes flat and does not remove material. This phenomenon is known as wheel wear. The wheel wear leads to changes in grinding conditions and loss of surface quality of the workpiece. Thus, with the loss of the cutting ability of the wheel, grinding forces and, consequently, the power consumption in the grinding head increase. Therefore, when the wheel is worn it is necessary to "bring to light" the abrasive grains and provide to the grinding wheel, again, its cutting ability. For this purpose, a dresser is used.

The wheel wear is usually represented by a curve with three distinct stages (Figure 5). Thus, in the initial part, the effect of dresser is very important (Chen, et al., 1998). The next stage is lineal and slightly incremental, and in this stage the effect of the dresser disappears. Therefore, in this area, only the wheel and the grinding conditions have influence over the wheel wear. This stage is usually used to characterize the wheel wear (Malkin & Guo, 2008). Finally, in the last stage the wheel wear increases exponentially. Thus, it is recommended to not enter in this zone when grinding and dress the wheel at the end of the second stage.

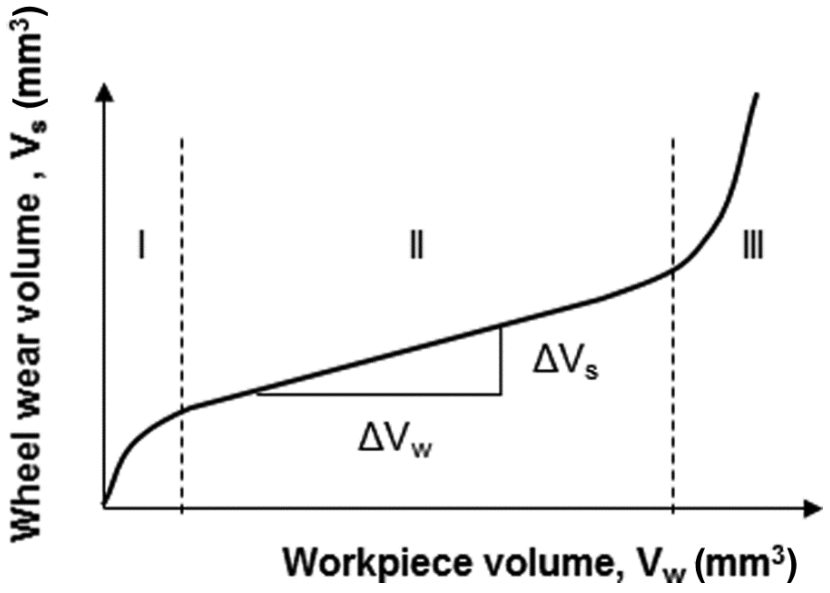


Figure 5 Grinding wheel wear typical curve (Linke, 2015)

When the wheel wear increases, as said before, the cutting capability of the wheel decreases and the surface roughness increases. The surface roughness refers to the surface finish quality of the workpiece. Typically, to measure the surface roughness of a workpice the following are used (Marinescu, et al., 2006):

- R_t: Is the SI parameter for maximum surface roughness. It measures the maximum difference between peak and valley distance within the sampling length.
- R_a: Is the arithmetic average height of roughness-component irregularities (peak, heights and valleys) from the mean line within a sampling length after filtering out form deviations.
- R_z: Is the arithmetic average of maximum vertical distance from the peak height to valley depth over five adjacent individual samplings lengths.

Besides the wheel wear and surface roughness, another key variable in grinding process is the specific grinding energy. It measures the amount of energy required to remove the unit volume of part material. Thus, it is significant because it gives information about mechanism and degree of contact between the abrasive and the workpiece. Besides, the specific grinding energy is also useful for estimating the power requirement of the grinding machine (Malkin & Guo, 2008).

2.2 Modelling the process

In 2009 the European Research area of the European Commission published a document about the strategy for a sustainable European machine tools industry (European Commission, 2009). In that document the research in the machine tools industry is divided into five sub-projects. One of them is "The Manufacturing Breakthrough" and it is said that "In this case four demonstrators are envisaged, illustrating self-calibration, predictive maintenance, versatile configuration, and improved control of accuracy and acceleration at high operating speeds."

Besides, by the time a part reaches a grinding machine, mostly, that part has undertaken significant machining already. Thus, if something should happen during the grinding, then all of that previous machining investment is lost. Therefore, it is highly important in grinding to safeguard the value that has already been added to the part (Zelinski, 2013). Besides, it must be taken into account that in the past decade, about 20%–25% of the total cost of all machining processes was due to grinding (Malkin & Guo, 2008).

Therefore, in grinding process monitoring and control are of primary importance in order to improve the accuracy and save time investment. However, in order to control a process like the grinding process, reliable and accurate models are needed. Therefore, one of the

main tasks before controlling the grinding process is to generate accurate models of the process.

For modelling the grinding process, two research lines can be highlighted, the classic approach, aimed at the scientific knowledge of the process; and the approach based on intelligent systems, capable of combining previous experience and knowledge based advanced models.

2.2.1 Theoretical models

The classical approach is based on analytical models. These models are mathematical expressions for predicting one or various process outputs from the different process variables. Thus, Marinescu et al. (Marinescu, et al., 2006) collected a number of different models for the different outputs of the grinding process. Among them, Equations (1) and (2) allow relating specific grinding energy (e_c), surface roughness (R) and grinding wheel wear (V_s):

$$e_c \approx k \cdot \sqrt{\frac{V_s}{V_w} \cdot C \cdot r \cdot \sqrt{\frac{d_e}{a_e}}} \quad (1)$$

$$R \approx \left(\frac{V_s}{V_w} \cdot \frac{1}{C \cdot r \cdot \sqrt{d_e}} \right)^{2/3} \quad (2)$$

Equations (1) and (2) show the non-linear dependency of specific grinding energy and surface roughness with grinding process parameters such as speed ratio, depth of cut, wheel diameter, and with the wheel wear, expressed through the product $C \cdot r$, where C is grain density and r is the so-called grit shape factor. Commonly, $C \cdot r$ can be considered as a single factor related to the surface topography and surface wear of the grinding wheel.

However, industrial application of the analytical models is not an easy task yet. The reasons for the limitations of theoretical explicit models for grinding wheel wear, workpiece surface roughness and specific grinding energy are the following: First, the lack of precise information about the composition and performance of the wheel itself due to the semi-handmade production of the grinding wheels. Then, as shown before, the relations between the different process variables and process outputs are highly non-linear. Finally, must also be taken into account the accuracy required in the final results. Besides, theoretical models need calibration before using them in practical grinding operations. All these facts have led many researchers to use intelligent techniques in order to model the grinding process.

2.2.2 Modelling using Artificial Intelligence

Using intelligent techniques for the estimation and/or prediction of outputs of the grinding process is a well-known approach. In most of the cases the output variable is surface roughness, although there are a number of research works in which wheel wear was studied. However, little effort has been dedicated to the prediction of specific grinding energy even though it is a fundamental grinding variable.

Research work	*Grinding wheels*	*Signals (Inputs)*	*Grinding conditions*	*Prediction algorithm*
(Nandi & Pratihar, 2004)	D126K5V	No input signals	34 different grinding conditions	FLC+GA
(Sedighi & Afshari, 2010)	Aluminium oxide	No input signals	16 different grinding conditions	ANN+GA
(Yang & Jin, 2010)	Not mentioned	Vibrations + Temperature	Fixed	ANN+GA
(Li & Liu, 2011)	GB70RAP	No input signals	18 different grinding conditions	ANN+GA
(Aguiar, et al., 2008)	38A80PVH	Electric Power + Acoustic emission (AE)	15 different depth of cuts	ANN
(Prabhu, et al., 2015)	10µm grit size vitrified alumina	No input signals	8 different grinding conditions	ANN and Fuzzy Logic
(Chandrasekaran & Devarasiddappa, 2014)	White aluminium oxide	No input signals	25 different grinding conditions	ANN

Table 1 Summary of previous research on surface roughness modelling on grinding

In Table 1 the summary of the previous research works that tried to model the surface roughness are shown. A Fuzzy Logic controller (FLC) was designed in (Nandi & Pratihar, 2004) for predicting the final surface finish and the required spindle power in the grinding of steel using a grinding wheel as given by the standard specification D126K5V. Thus, the Fuzzy Logic controller had four cutting parameters as inputs in order to predict both variables. First, Genetic Algorithms (GA) were used to optimize the membership function distributions of the variables and as well as rule base. At the end, the system was able to predict a unique surface roughness and power for the given inputs.

For another specific type of grinding operation, the so-called creep-feed grinding, ANNs were used for the prediction of surface finish and maximizing the material removal rate

(Sedighi & Afshari, 2010). First, the ANN was used for predicting the surface roughness with alumina grinding wheel. The network inputs were four grinding cutting condition parameters. Then, once surface finish is known. Genetic Algorithms optimization techniques were used for the maximization of material removal rate and the minimization of surface roughness.

Genetic Algorithm techniques were also used for training an ANN for modelling surface roughness, precisely, for predicting the peak to valley value in surface grinding (Yang & Jin, 2010). The network inputs were the workpiece and wheel temperature on one hand, and the vibration of the spindle wheel and workpiece on the other. Thus, an infrared image camera and a vibration sensor were used to collect temperature and vibration signals, respectively. No mention was made about the grinding wheel used to carry out the experiments.

In a more recent work, a similar approach was presented (Li & Liu, 2011) for modelling the surface finish in cylindrical grinding of steel parts using a GB70RAP400 grinding wheel based on a Malkin surface roughness model. In this case, four inputs such as workpiece speed and traverse speed were used to characterize the grinding operation. It is important to note that a reference was done about the on-line measurement of some variables such as displacement and speed of the worktable. However, during the training only three discrete values are used for each network input. Besides, the network was trained using Genetic Algorithms optimization method.

Also ANNs were used in (Aguiar, et al., 2008) for modelling the surface roughness for different depth cut for steel parts grinding with 38A80PVH grinding wheel. In this work, a Hall-effect transducer was used to measure the electrical power consumed from the three-phase induction motor that drives the wheel wear. Besides, an Acoustic Emissions (AE) sensor was also used close to the workpiece. However, no reference was made about the reason of placing the AE sensor close to the workpiece which was it duty. Then, different ANN configurations were tested with different inputs related with AE signal, power signal and grinding conditions.

In the following works ANNs were also used for predicting the surface roughness for plane grinding with 10μm grit size alumina grinding wheel (Prabhu, et al., 2015). Thus, three machining parameters were used as inputs of the net. Besides, the capability of prediction of the surface roughness by ANNs was compared with a fuzzy logic based solution. The results show that the performance of the ANN was slightly better.

Again an ANN was used in cylindrical grinding for modelling the surface roughness using white aluminium oxide grinding wheel (Chandrasekaran & Devarasiddappa, 2014). Four input machining parameters at three different levels were considered for experimentation. Thus, combining these inputs and levels 20 training experiments and 5 testing experiments were generated. The network was able to predict the surface roughness.

Research work	*Grinding wheels*	*Signals (Inputs)*	*Prediction algorithm*	*Output*
(Lezanski, 2001)	38A80KVBE	AE + Vibrations	ANN+Fuzzy	sharp or dull wheel
(Liao, Hua, Qu & Blau, 2007)	SD220R75B56-1/8 and SD220R100B99-1/8	AE	Hidden Model of Markov	good or bad condition
(Liao, Hua, Qu & Blau, 2008)	SD220R75B56-1/8 and SD220R100B99-1/8	AE	AdaBoost and A-Boost	sharp or dull wheel
(Yang & Yu, 2012)	WA60LmV	AE	Support Vector Machines	sharp or dull wheel
(Hosokawa, et al., 2004)	CBNC140N75B	Sound	ANN	4 wheel conditions
(Maksoud & Atia, 2003)	4 grinding wheels	Piezoelectric + high response pressure transducer	ANN	sharp or dull wheel

Table 2 Summary of previous research on Wheel wear modelling on grinding

On the other hand, when it comes modelling the wheel wear, nearly all research works that use Intelligent Techniques are devoted to distinguish between a sharp and a dull wheel (Table 2). In (Lezanski, 2001) a neuro-fuzzy system for cylindrical grinding for a 38A80KVBE grinding wheel is described. The ANN was used for reducing the number of inputs to the system. The selected inputs were machining parameters and parameters extracted from AE and vibration signals. Then, the fuzzy system was applied to decide whether the wheel was in sharp or in dull (worn) condition.

In (Liao, et al., 2007) Hidden Model of Markov (HMM) was used to predict the state of SD220R75B56-1/8 and SD220R100B99-1/8 diamond grinding wheels during creep-feed

grinding. In this work also AE signal was used for clustering the grinding wheel state between good or poor condition.

In a later work (Liao, et al., 2008), the research was completed by using automatic boosted classifier, AdaBoost and A-Boost precisely, to detect online a dull wheel based on AE signals. The same SD220R75B56-1/8 and SD220R100B99-1/8 diamond grinding wheels were also used in this work.

In another work, Support Vector Machines (SVM) were proposed as classification method between dull and sharp for white fused alumina WA60LmV wheel (Yang & Yu, 2012). Also AE signal is used in this work. After, signal pre-processing and future extraction SVM was used to classify between sharp and worn state. Besides, genetic algorithms (GA) were introduced to select the optimal parameters automatically. Finally, comparison between SVM-GA system and ANN was done for wheel state classification. The results show that the classification accuracy of SVM-GA system was higher.

In (Hosokawa, et al., 2004) ANNs were used for modelling the status of the A60K7V and CBNC140N75B grinding wheels. The main objective was to analyse the effect of the dressing over the wheel wear. Unlike in other works, in this case grinding sound emitted from wheel surface was used. The proposed ANN had four outputs, thus, using the signal sound, the network was capable to classify between different grinding wheel conditions instead of only worn or dull.

In (Maksoud & Atia, 2003), a methodology for implementing a grinding process controller for surface grinding based on neural networks was presented. The controller was capable to decide when to dress the grinding wheel before exceeding the surface roughness limit. Firstly, one network was developed for designing a grinding process based on the grinding wheel, workpiece and desired surface roughness. Actually, this network predicted the machining parameters to achieve the desired surface roughness. Secondly, using some sensors for measuring the grinding forces (Piezoelectric transducer) and wheel topography (high response pressure transducer), a second network was designed and used for controlling the process and deciding when to dress the wheel. Four different grinding wheels were used. However, the wheels used for training and testing were the same ones, so as only the machining conditions changed.

2.3 Conclusions

Machining tool processes are nowadays key technologies in a competitive global manufacturing marketplace. Despite of the advances in the performance and accuracy of other machining tool processes for manufacturing, such as turning or milling, grinding process has become key in manufacturing. Actually, due to its capacity for producing parts of high precision and high surface quality in difficult-to-machine materials, grinding is widely used in high-added value sectors such as motor, aerospace industries or precision cutting tool manufacturing.

In grinding process control and monitoring are of primary importance in order to improve the accuracy and save time investment. However, in order to control a process like the grinding process, reliable and accurate models are needed. Therefore, one of the main tasks before controlling the grinding process is to generate accurate models of the process.

For modelling the grinding process, two main research lines can be highlighted, the theoretical models approach, aimed at the scientific knowledge of the process; and the approach based on intelligent systems, capable of combining previous experience and knowledge based advanced models.

However, industrial application of the theoretical models is not an easy task yet. The reasons for the limitations of theoretical explicit models are the lack of precise information about the composition and performance of the wheel itself due to the semi-handmade production of the grinding wheels, and the relations between the different process variables and process outputs are highly non-linear. Besides, theoretical models need calibration before using them in practical grinding operations. Therefore, many researchers use intelligent techniques in order to model the grinding process.

Almost all the research works that use intelligent techniques found in the scientific literature provide particular solutions for a given wheel-workpiece pair. In few cases more than one grinding wheel or workpiece are used. Results cannot be generalized, in no case, to other types of grinding wheels not used during the design of the models. In fact, in some of the cases, they cannot even be generalized to new grinding conditions that have not been used during the training of the model. This is one of the reasons why industrial application of intelligent techniques in grinding has been very limited so far. Besides, in the cases of prediction of surface roughness, the output of the model is the value related to the current state of wear of the grinding wheel being this value considered unique for

the whole grinding process. However, as said in Section 2.1 the surface roughness of the ground components changes over time as the wheel loses its cutting ability.

Likewise, all the works that model the wheel wear use signals like Acoustic Emissions (AE), vibration signals or audio signals. However, accessibility of the machining area is limited due to the very aggressive conditions and low accessibility to it. Actually, the high rotational speed of the grinding wheel (common values are around 45 m/s, but with industrial examples with values up to 200 m/s), the generation of abrasive swarf or the presence of large quantities of coolant limits the possibility of using industrial sensors during the process in the machining area. Besides, in most of the cases, the given solution is binary, the dull or sharp condition of the wheel. Thus, the solutions do not give the actual status of the grinding wheel. In fact, the designer decides when the grinding wheel is dull or sharp before developing the intelligent model.

Finally, as said in 2.2.2, little effort has been dedicated to the modelling of the specific grinding energy with intelligent techniques. However, it is a fundamental variable in order to know the performance of the grinding process and it is also useful for estimating the power requirement of the grinding machine.

3 MODELLING DYNAMIC EVOLUTIONS WITH ANN

Unlike the works found in scientific literature, in this work the wheel wear, the surface roughness and the specific grinding energy are considered as dynamic evolutions. Likewise, it would be necessary to generalize to new grinding wheels and grinding conditions in order to break the wheel-workpiece pair. ANNs characteristics made them suitable for modelling highly no-linear systems like grinding process. ANNs refer to computing systems whose central theme is borrowed from the analogy of biological neural networks. Artificial Neural Networks are well known for dynamic evolutions modelling due to their capability to learn the relationship between the inputs and outputs of the system. In most of the cases ANNs are used for modelling dynamic systems called time series, based on the historical data. However, in some cases the past or historical data is not available because measuring real values is time and resource consuming. In those cases, the aim is to predict a complete dynamic evolution but without initial or real values. Therefore, in order to model the specific grinding energy, wheel wear and surface roughness, it is crucial to predict a complete dynamic evolution without measuring any initial real value. Besides, it is reasonable to think that one ANN will not be capable to model all the grinding wheels and grinding conditions. Thus, a methodology for generating new ANNs for different application fields should be developed with generalization capabilities concerning new grinding wheels and new grinding conditions.

Although intelligent models are also far from being used in industrial environments, the literature review reveals that Intelligent Techniques and, more precisely, ANNs constitute, in fact, an effective approach for modelling the grinding process. The following ANNs characteristics make them suitable for modelling highly no-linear systems like grinding process (Isasi & Galván, 2004):

- Model highly no-linear and stochastic systems.
- ANNs can learn from examples. Thus, for developing ANN-based models real experiments are used.
- High generalization capability.

Besides, artificial neural networks also provide one important advantage:

- For the construction of ANN-based models analytical expressions of underlying physical phenomena are not necessary. Actually, the neural network models are built automatically through a training procedure based on data from experiments (Haykin, 1998).

Therefore, the objective of this work is to model the specific grinding energy, wheel wear and surface roughness with ANNs. But unlike the works found in scientific literature, in this work the wheel wear, the surface roughness and the specific grinding energy are considered as dynamic evolutions with an initial point (the wheel is sharp) and final point (the wheel is dull). Besides, it would be necessary to generalize to new grinding wheels and grinding conditions in order to break the wheel-workpiece pair and develop models suitable for industrial applications.

Thus, this chapter deals with the description of the basic theory of the artificial neural networks and how have been used for modelling dynamic evolutions. Thus, in the first section, a basic theory about artificial neural networks, artificial neural network architectures, activation functions and learning is presented. The aim of this chapter it is not to give a deep description of all the aspects involved. Actually, the objective is to introduce the reader into ANNs. In fact, if the reader wants to deepen more into artificial neural networks theory, it would be recommended reading references provided.

On the other hand, after the ANNs basic theory description, the use of neural networks for modelling dynamic systems is shown. Briefly, the ANNs are mostly used to model dynamic evolutions based on historical values. However, in some cases the dynamic evolutions are modelled using initial real values or another dynamic evolution. A extended analysis is provided in Section 3.2.

3.1 Artificial neural networks

The Encyclopædia Britannica defines the human intelligence as ‘the mental quality that consists of the abilities to learn from experience, adapt to new situations, understand and

handle abstract concepts, and use knowledge to manipulate one's environment'. For a long time ago, one of the major objectives of the science has been giving the human intelligence to the machines. Thus, this has led to the study and simulation of human neural networks i.e. the brain.

3.1.1 Biological neuron

The human brain consists of a large number of interconnected neurons. Each of these neurons consists of dendrites, the cell body and the axon (Figure 6). The dendrites receive and carry the electricity signals into the cell body. Furthermore, the axon carries the signal from the cell body out to other neurons (Hagan, et al., 2014). The contact-point between an axon, which end extends in a tree, of one cell and a dendrite of another cell is called a synapse, across which information is propagated.

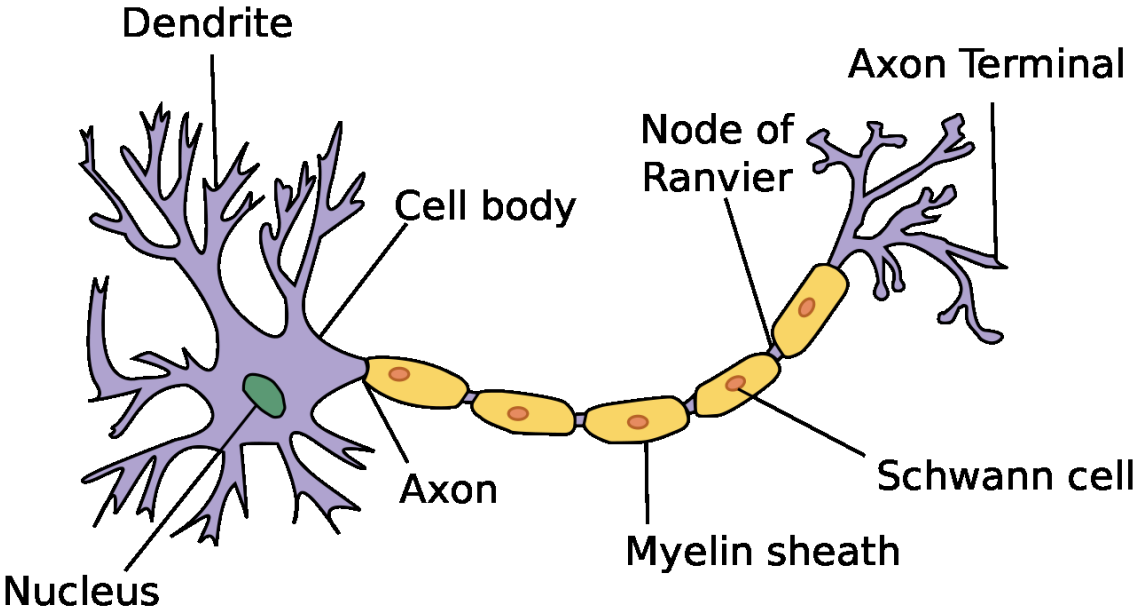


Figure 6 Graphical representation of a biological neuron (https://commons.wikimedia.org)

The magnitude of the signal received by a neuron from another one depends on the strength of the connection between the neurons, i.e. the strength of the synapsis. The cell membrane becomes electrically active when it is sufficiently excited by the neurons making synapses onto this neuron. A neuron will send an output impulse if sufficient signals from other neurons fall upon its dendrites in a short period of time. The neuron fires if its net excitation exceeds its inhibition threshold. Firing is followed by a brief *refractory period* during which the neuron is inactive (Mehrotra, et al., 1997).

Biological neurons are very slow when compared to electrical circuits (Hagan, et al., 2014). However, biological neural networks are much more complex than Artificial

Neural Networks (ANNs). Actually, it is estimated that each neuron is connected with other 10000 neurons, creating a huge network working in parallel, operating at the same time. Therefore, even each neuron has low processing capacity, a large number of them interconnected and working in parallel can perform any task much faster than any modern computer.

3.1.2 Artificial neuron

ANNs refer to computing systems whose central theme is borrowed from the analogy of biological neural networks (Mehrotra, et al., 1997). Thus, as in the human brain, the ANN is a highly parallel distributed network of neurons. The neuron is the basic processing unit of the neural network. In Figure 7 it is shown the block diagram of a neuron. The basics of the neuron are as follows (Haykin, 1998):

- Weights: Represent the synaptic strengths of the neuron. Thereby, an input x_j of weight j connected to neuron k is multiplied by the weight w_{kj}. Unlike the synapses of the human brain, the weights of ANNs have negative or positive values.
- Adder: A linear combiner of the weighted inputs.
- Activation function: For limiting the amplitude of the output of a neuron. Normally, this amplitude within the normalized ranges of [0, 1] or [-1, 1].
- Bias: It increases or decreases the input to the activation function, depending on the positive or negative sign of the bias (b_k).

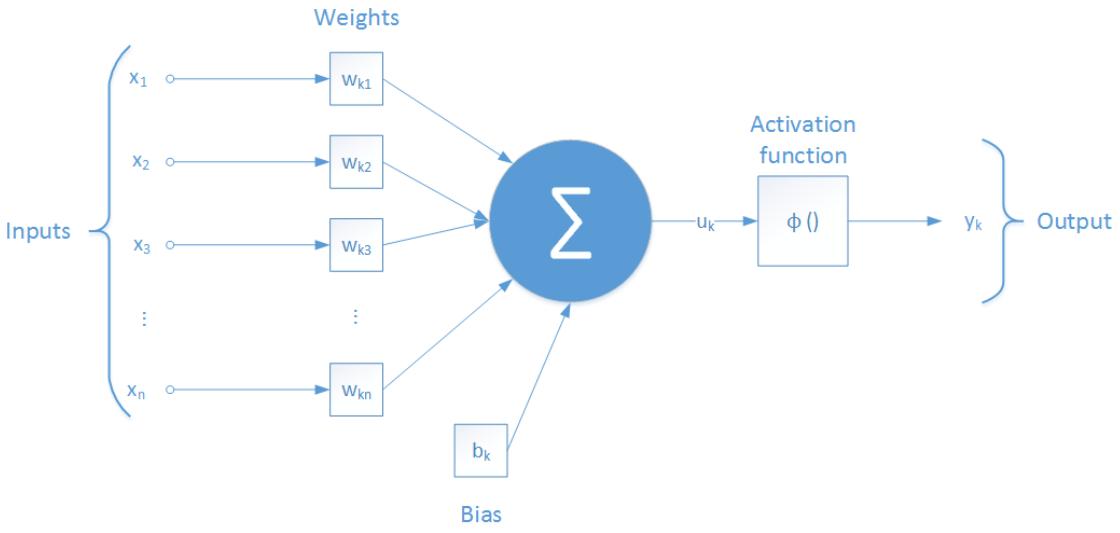


Figure 7 The block diagram of a nonlinear model of a neuron

The block diagram of a neuron represented in Figure 7 can be expressed mathematically with the following two equations:

$$u_k = \sum_{j=1}^{n}(w_{kj}x_j) + b_k \quad (3)$$

$$y_k = \varphi(u_k) \quad (4)$$

where $x_1, x_2, x_3, \ldots, x_n$ are the inputs to a neuron; $w_{k1}, w_{k2}, w_{k3}, \ldots, w_{kn}$ are the weights of a neuron; u_k is the linear combination of the weighted inputs; b_k is the bias of a neuron; $\varphi()$ is the activation function; and y_k is the output of the neuron.

3.1.3 ANNs architectures

Rarely one neuron is sufficient to model the complex relationship between the inputs and the outputs of the neuron (Hagan, et al., 2014). Thus, usually more than one neuron working in parallel is needed. These neurons working in parallel constitute a 'layer'. Thus, combining neurons in one or more layers' new network architectures can be designed to solve a specification problem.

3.1.3.1 Single-neuron neural networks

This is the simplest architecture of layered neural networks. It only has one computation layer, the output one. Actually, the input layer is not considered because no computation is done there. It only consists of source nodes that project onto the output layer. It is important to appoint that the data evolve in one direction, from inputs to outputs. Actually, these networks are usually called single-neuron *feedforward* neural networks. An illustration can be seen in Figure 8.

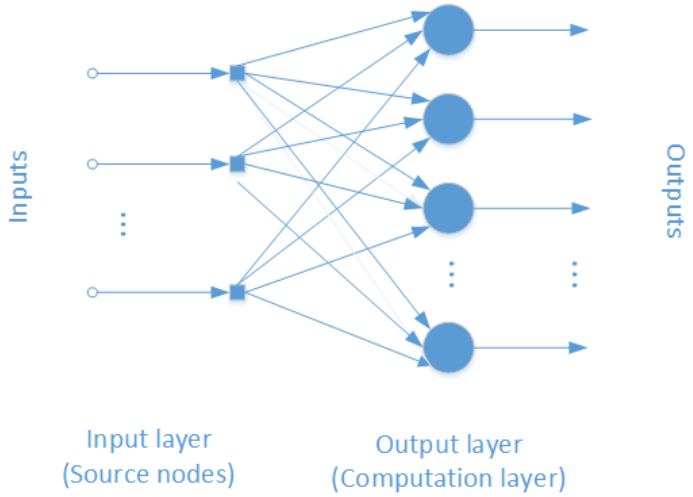


Figure 8 Single-layer neural network

3.1.3.2 Feedforward Neural networks

To model more complex relationships one or more hidden layers are introduced between the input and output layer. The computation nodes of these layers are called hidden units or neurons. Thus, the inputs project onto the first hidden layer, then, the output of this layer are used as inputs of the second hidden layer and so on for the rest of the layers. In the same way as the single layer neural networks, the data goes in one way, from the input to the output layer passing through the hidden layers.

In Figure 9 a one hidden layer neural network is shown. The network has 3 inputs, 5 neurons in the hidden layer and 2 output neurons (one for each output signal). Thus, for simplicity, this network is referred to as 3-5-2 network. Besides, the network in Figure 9 is said that it is fully connected because every node in each layer is connected to every node in the adjacent layer (Haykin, 1998).

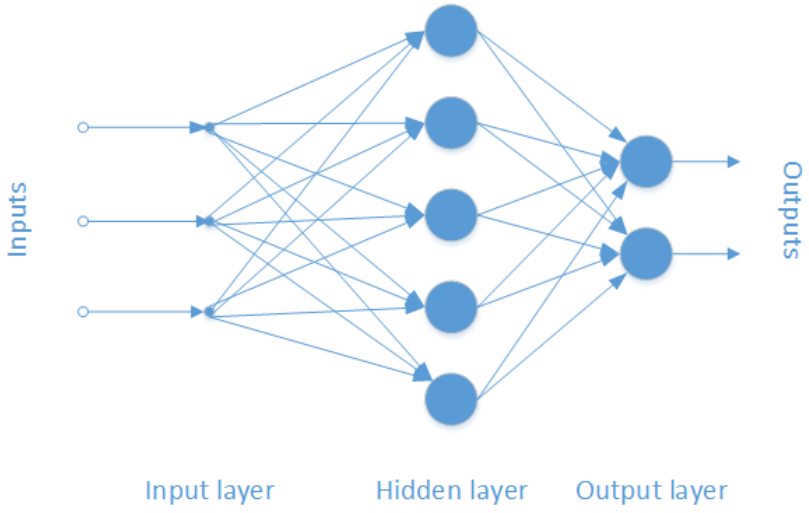


Figure 9 Three-layer neural network

The most common neural networks are the Multi-Layer Perceptron (MLP). The MLP is a generalization of the single-layer neural network considered in Section 3.1.3.1. Multi-layer perceptron neural networks are more powerful than single-layer ones. Actually, a trained one hidden layer network having sigmoid activation function in the hidden layer and linear activation in the output layer can approximate most functions (Hagan, et al., 2014). In fact, the hidden layers neurons with sigmoid or hyperbolic tangent functions adds a smooth nonlinearity to the network (Haykin, 1998).

It is noteworthy to appoint that is possible to model dynamic evolutions with feedforward neural networks. In fact, they are widely used. For that, usually in the input of the network

lagged inputs are used. Thus, the network is able to model the dynamic behaviour of the evolution (see Section 3.2.1).

3.1.3.3 Recurrent Neural Networks

In recurrent neural networks the data do not go only feedforward, from the input layer to the output layer. At least, it has one feedback loop. These feedback loops have delay units (Figure 10). Thus, the feedback loops have a high impact on the training and performance of the neural network (Haykin, 1998). Depending on the source and direction of the feedback loop different recurrent neural network topologies can be developed.

For instance, talking about the *classical* recurrent neural networks, in the well-known Elman neural networks (Elman, 1990) the feedback loop goes from the output of the hidden layer to the input of the hidden layer. In another widely used recurrent network proposed by Jordan, the feedback loop goes from the output of the net to the input of the hidden layer (Jordan, 1990). In the case of the Nonlinear AutoregRessive eXogenous model (NARX) it has a feedback from the output of the network to the input, like the Jordan network, but it also has delayed inputs. Besides *classical* recurrent neural networks, other networks can be found in the literature such as Echo State Networks (ESN) (Mass, et al., 2002) based on reservoir computing or Long Short-Term Memory (LSTM) networks for learning long-term dependencies and to avoid the gradient descent vanish (Hochreiter & Schmidhuber, 1997).

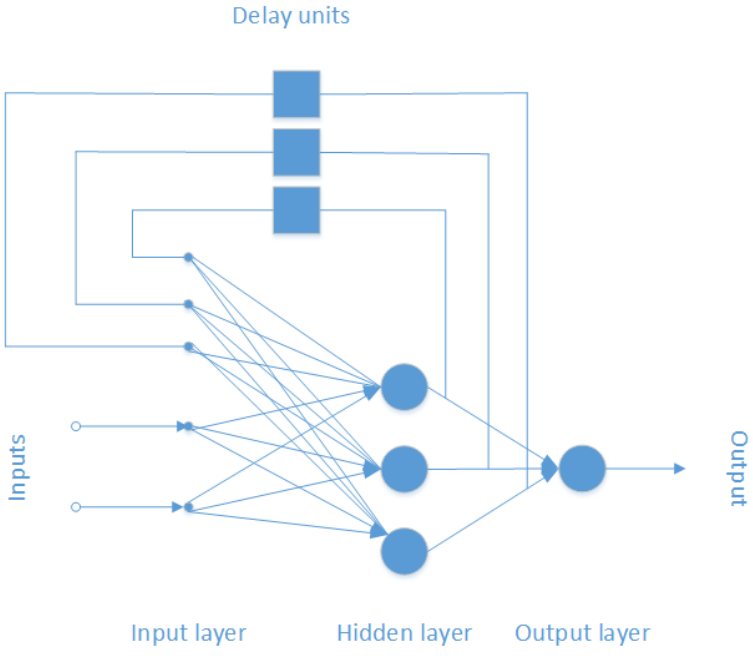


Figure 10 Recurrent neural network: an example

3.1.4 Activation function

The activation function defines the output of a neuron in terms of the induced local field (Haykin, 1998). Normally, a particular transfer function is chosen by the designer to solve a specification problem (Hagan, et al., 2014). The most used activation functions are the following:

1. *Threshold function:* The threshold function is defined by the following mathematical expression:

$$\varphi(x) = \begin{cases} 1 & if\ x \geq 0 \\ 0 & if\ x < 0 \end{cases} \quad (5)$$

In this model the output of the neuron is 1 if the the linear combiner of the weighted inputs is nonnegative. Thus, the output value of the neuron is 0 or 1.

2. *Linear function:* For the linear function:

$$\varphi(x) = x \quad (6)$$

In the linear function the output value takes the same value of the induced local field. It is normally used in output neurons of artificial neural networks.

3. *Sigmoid function:* For this type of functions we have:

$$\varphi(x) = \frac{1}{1+e^{-x}} \quad (7)$$

One of the most used activation functions in artificial neural networks. It is an increasing function that exhibits the goodness of the linear and nonlinear behaviour (Haykin, 1998). The output of the net yields within the range [0, 1] and, hence, only takes positive values.

4. *Hyperbolic tangent function:* The hyperbolic tangent function is defined by the following mathematical expression:

$$\varphi(x) = \frac{1-e^{-x}}{1+e^{-x}} \quad (8)$$

It is similar to the sigmoid function and its graph is also a shaped S. However, with hyperbolic tangent function the output of the neuron can take positive or negative values and the output value range is [-1, 1]. This activation function is, also, one of the most used in the construction of an ANN.

3.1.5 Learning

One of the main characteristics of the artificial neural networks is their capacity to *learn* from examples. Actually, the ANN improves its performance thanks to the *learning*. During the *learning* phase of the net the weights and biases are adjusted in an iterative

process, also, called training phase. The learning or training process implies the following sequence of events (Haykin, 1998):

- The ANN is *stimulated* by the environment.
- As the stimulation of the environment ANN parameters (weights and biases) are changed.
- Due to changes in the neural network, it responds in a new way to the environment.

3.1.5.1 Learning paradigm

In the learning process of the ANNs three basic learning paradigms can be found: supervised, reinforcement and unsupervised learning (Haykin, 1998).

In the supervised learning the desired output values are shown to the network during the training. Actually, it can be said that a "teacher" provides "examples" of inputs and corresponding output(s) (Dreyfus, 2005). Thus, the training algorithm adjusts the net weights and biases to reduce the error between the desired and predicted outputs.

The reinforcement learning is quite similar to the supervised learning (Hagan, et al., 2014). However, instead of showing the corresponding output(s), the "teacher" gives a grade (or score) of the performance of the net for the provided "examples" of inputs.

On the other hand, unlike in supervised learning, in unsupervised learning there is not a "teacher" to guide the learning process. Thus, during the training process the net should find the similarities between the elements of the inputs and, thus, translate these similarities into neighbourhoods in the new data representation (Dreyfus, 2005). Therefore, this learning method is usually applied to data clustering or image recognition.

3.1.5.2 Training algorithm

As commented above, during the *learning* phase of the net the weights and biases are adjusted in an iterative process, also, called training phase.

Thus, a set of rules that follow these events are called training algorithm or learning algorithm. There are several training algorithms that differ from each other in how they interconnect with the examples or how they adjust the net parameters.

The most common training algorithm for multi-layer perceptron neural networks is the backpropagation (BP) training algorithm under supervised paradigm. In fact, sometimes MLP networks are denoted as backpropagation neural networks. This training algorithm works under the supervised learning paradigm. The backpropagation is divided in two

phases: forward and backward phase (Haykin, 1998). In the forward phase the input "examples" propagate from the input layer though layer by layer to the output, producing a response. The produced response is compared with the target or desired output response. Then, the resulting error comparing the output and desired responses is propagated backward from the output trough layer by layer to the input. In this backward phase, the network weights and biases are adjusted to minimize the sum of squared errors. In fact, the BP is a gradient descent algorithm where the gradients are propagated backward.

3.1.5.3 Batch Vs. Incremental

As said before, the main objective of the learning phase is to update de weights and biases in order to learn the relationship between the inputs and the outputs of the network. However, for updating these weights under supervised learning two approaches can be found related to when to update the weights: batch and incremental. In batch training the weights and biases are updated only after all training samples are presented to the network. In incremental training, the weights and biases are updated after every sample presentation (Mehrotra, et al., 1997).

3.1.5.4 Generalization

In ANNs generalization refers to the capacity of the networks to map the input-output relationship for new data not used during the training process of the network. Generalization is the opposite to the overfitting of the net that occurs when the network learns from the training examples but it is not capable to generalize to new examples. An ANN that is designed with capabilities to generalize can map the input-output relationship of examples slightly different of the examples used to train the network (Haykin, 1998). Therefore, to avoid the overfitting and improve the generalization of the net different strategies have been addressed such as early-stopping, cross validation or regularization.

3.2 Modelling dynamic evolutions with ANN

On the real world most of the systems have a dynamic behaviour, where actual states depend on past events or actions. Thus, modelling dynamic systems is a challenging task for predicting future events by analysing the past in a wide range of fields, such as weather, finance, earthquakes or machining tools wear. These systems can be represented as dynamic evolutions of collected observations. These observations might be continuous

or in discrete intervals, fixed or not. The most common dynamic evolutions have fixed discrete intervals.

One of the most common dynamic evolutions are the time series. In time series, the fixed discrete intervals are related with time such as hour, days, months or years. However, some dynamic evolutions are not measured in time, in fact, it could be said that they are pseudo-time series. Thus, in this kind of dynamic evolutions the data evolve based on parameters not related with time. For example, in machining, tools wear or live-cycle are measures related to the use.

Artificial Neural Networks are well known for dynamic evolutions modelling due to their capability to learn the relationship between the inputs and outputs of the system (Štěpnička, et al., 2013), learning by examples (Cukrowska, et al., 2001) and model non-linear systems. ANNs used for modelling dynamic evolutions can be classified into two main categories: feedforward neural networks like multilayer perceptron (MLP) or radial basis function (RBFN) networks, where all data go forward from de input to the output; and recurrent neural networks (RNN) like Elman or Jordan networks, where there is data feedback.

At this point, it would be important to explain two concepts that are used in dynamic evolutions modelling with ANN. These concepts are one-step ahead and multi-step ahead. In Figure 11 a chart of both strategies are shown. In the chart the dashed signal represents the target signal, the signal to predict. On the other hand, the arrows represent the predictions (O_n), and the dotted lines indicate the prediction error (e_n). Thus, it can be seen that in one-step ahead the prediction always starts from the real value i.e. the $t+1$ value is predicted using at least t real values. On the other hand, in multi-step ahead the prediction horizon increases. Thus, $t+2$ value prediction is based not only on the t real value, but also on $t+1$ predicted value.

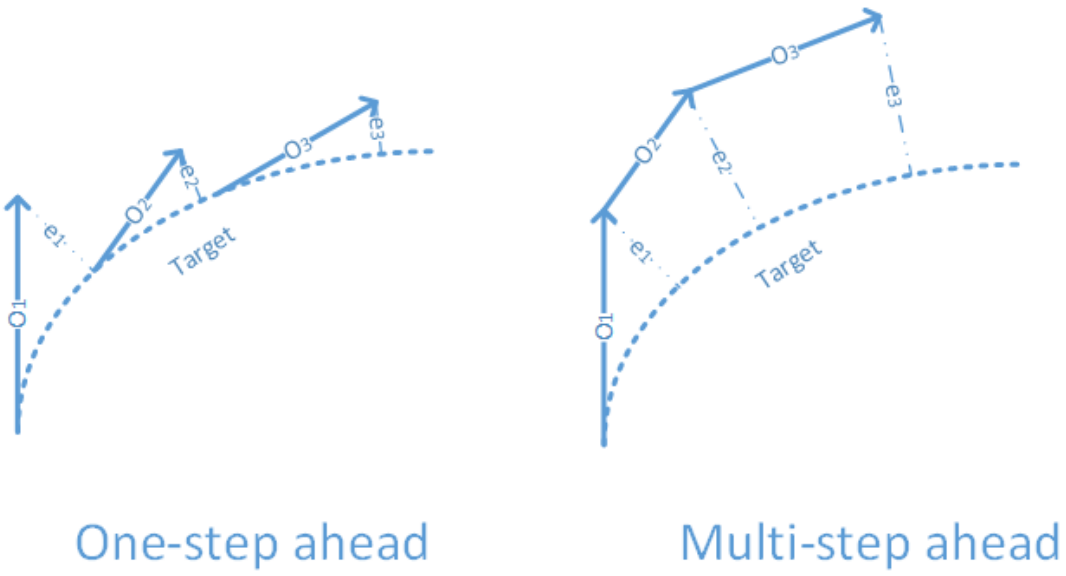


Figure 11 One-step ahead Vs. Multi-step ahead prediction (Bakker, Schouten, Giles, Takens, & van den Bleek, 2000)

Therefore, the multi-step ahead prediction is more changeling because the prediction error increases at each prediction step. However, the multi-step ahead prediction has more potential because, as said before, the prediction horizon increases. Thus, it is possible to predict more time-steps ahead. For example, in weather forecasting, with one-step strategy, tomorrow's weather is predicted. However, with multi-step ahead it is possible to predict day the after tomorrow's weather or all week's weather.

3.2.1 Modelling dynamic evolutions using feedforward neural networks

Feedforward neural network does not take into account the past values of the dynamic evolutions, however, some feedforward architectures use lagged inputs (*N-1, N-2…, I-1, I-2…*) in the input layer of the ANN for predicting future values, and/or a moving buffer where the predicted value becomes the input in the next step.

Thus, in (Claveira & Torra, 2014) the authors use the well-known MLP in order to forecast the tourism demand in Catalonia. To model the dynamic evolution of tourism the used net had three layers trained with the Levenberg-Marquardt backpropagation algorithm. In the article the authors do not specify the number of the neurons in the hidden layer. However, they compare the ANN forecasting performance for 1, 2, 3, 6 and 12 months ahead to AutoRegressive Integrated Moving Average (ARIMA) and Self-Exciting Threshold AutoRegressive (SETAR) models.

Monthly data of tourist arrivals and overnight stays from foreign countries over the time period from 2001 to 2009 was used in the comparative study. The ARIMA and SETAR models were estimated using the data from January 2001 to January 2008. Thus, the values from January 2009 to December 2009 were used to compute the forecast errors in a recursive way for the different forecast horizons.

On the other hand, the data was divided into three different datasets (training, validation and test) in order to avoid overfitting. Instead of using all data available, they use four-year period data divided into first fifty observations for training, the next twenty-one for validation and the rest 10% for testing the trained net.

The results showed that the ARIMA model outperforms the SETAR and ANN models, especially for shorter time horizons. Thus, the authors concluded that it would be necessary to pre-process the data because the seasonality can affect the prediction performance of the net. Besides, the authors appoint that the ANN could improve by adding input lags or memory values.

In another work the backpropagation neural network was also used in order to predict infectious diarrhea using meteorological factors (Wang, et al., 2015). The three layer network has nine inputs and one output (Table 3) while the neurons in the hidden layer were selected iteratively. Besides, the Levenberg-Marquardt was used as a training algorithm because, as the authors said, it had been used successfully to model many engineering applications.

For training the network 209 weeks data pair from 3 January 2005 to 4 January 2009 were used. This data was divided into two different datasets: the 80% was used to achieve the best network and the rest 20% to test the net. In this case, neither early-stopping neither Bayesian regularization were used, actually, the 5-fold cross validation was selected to avoid the overfitting.

Input (independent variables)	Parameters (Unit)	Symbol
Meteorological factors	Weekly average maximum temperature (ºC)	T_{max}
	Weekly average minimum temperature (ºC)	T_{min}
	Weekly average temperature (ºC)	T_{avg}
	Weekly average minimum relative humidity (%)	RH_{min}
	Weekly average relative humidity (%)	RH_{avg}
	Weekly average atmospheric pressure (hPa)	AP_{avg}
	Weekly average sunshine duration (h)	SD
	Weekly average wind speed (m/h)	WS_{avg}
	Weekly average rainfall (mm)	R_{avg}
Output (dependent variables)	Parameters (Unit)	*Symbol*
Infectious diarrhea	Weekly number of infectious diarrhea (case)	*WNID*

Table 3 Description of input and output parameters for constructing the prediction models (Wang, et al., 2015)

In order to decide the best network configuration a comparative study of the number of neurons in the hidden layer was carried out. For this, first, several heuristic approaches were used. However, the obtained results vary from 2 to 19 neurons in the hidden layer. Thus, the network was trained twenty times for each number of neurons. The results showed that the lowest validation RMSE was achieved with 4 neurons. Besides, the comparison between different activation functions (linear, sigmoid and hyperbolic tangent functions) with different normalization ranges (0-1, 0.1-0.9, 0.05-0.95 and 0.05-1) was made. The comparison results showed that the best results were yielded with a sigmoid activation functions in the hidden layer, linear function in the output layer and 0.05-0.95 normalization. However, it is important to notice that they did not use negative-positive range such as -1-1 or -0.95-0.95 with the hyperbolic tangent function. Finally, a

comparison study was made for different learning rates in order to analyse the influence over the weight and biases adjustment. The study determined that the learning rate value of 0.3 was the best one.

Using the best network configuration, it was trained 20 times and the yielded results were 40.367 root-mean-square error (RMSE) and 0.2556 mean absolute percentage error (MAPE). The performance of the net support vector regression (SVR), random forest regression (RFR) and multivariate linear regression (MLR) was also compared. The results clearly showed that their solution outperforms the other modelling approaches.

In another work (Ticknor, 2013), a three layered feedforward network was used for stock market forecasting. The network inputs were the daily stock data (low price, high price and opening data) and other six financial indicators that were not indicated in the paper. Thus, nine inputs, without lagged data, were used to predict the next day closing price of the chosen stock. Besides, the Levenberg-Marquardt backpropagation algorithm for training and the Bayesian regularization to improve the generalization of the network to new data were selected.

In this case, two stock during 734 trading days were collected, from 4 January 2010 to 31 December 2012. Thus, each sample consisted of daily low price, high price, opening price, close price and trading volume. The data was divided into two sets: 80% for training the net and the rest 20% for testing. It is important to highlight that for Bayesian regularization it is not necessary to have the validation dataset. Firstly, the net was trained to determine the number of hidden neurons in the hidden layer. Thus, the number of neurons was adjusted, at each iteration, until a constant number of effective parameters were found (Table 4). The network provides on average more than 98% fit to one-day ahead for both stocks.

Stocks	*Neurons*	*Layers*	*Effective number of parameters*
Microsoft (MSFT)	5	3	20.0
Goldman Sachs (GS)	5	3	20.2

Table 4 ANN parameters for stocks in initial experiment (Ticknor, 2013)

Besides, in the work the ANN performance was compared to Fusion model with weighted average (Hassan, et al., 2007) and ARIMA model. For the comparison study, other stock market values (Apple and IBM) from 10 February 2003 to 21 January 2003 were used. This dataset was divided as follows: from 10 February 2003 to 10 September 2004 for training, and from 13 September 2004 to 21 January 2005 for testing. The proposed ANN

with Bayesian regularization performance was as good as the other models. The authors' conclusion was that the addition or substitution of other technical indicators could improve the quality of the ANN model.

Likewise, for forecasting stock price, an ANN was used (Laboissiere, et al., 2015). However, in this case the objective was to predict the maximum and minimum stock prices of three Brazilian power distribution companies. One multi-layer perceptron neural network was used for each stock value maximum and minimum, six in total. The historical data considered begins in January 2008 and ends in September 2013. Before training the data, it was pre-processed in order to filter possible fluctuations and, thus, evidence trends. To do this, the Weighted Moving Average (WMA) was calculated for a window of 30 days.

From a bunch of 40 possible inputs, based on a correlation analysis between a possible network input and the output, those that had the highest correlation were selected. It is important to underline that they considered the previous days values up to five in some of the inputs. Thus, in some cases only the previous value was enough to predict the next day price. Nevertheless, in order cases, five days delayed inputs yielded the highest correlation value. Once the inputs were selected one and two hidden layers networks were considered. For one hidden layer 5, 10, 15, 20, 25 and 30 neurons were tested. For the case of two hidden layers 5, 10, 15, 20 and 25 neurons in the first hidden layer and 10, 15, 20, 25 and 30 neurons in the second hidden layer were assumed. The networks were trained using Levenberg-Marquardt training algorithm and the sigmoid function as the activation function. In the paper it is not mentioned how many times each configuration is trained.

The results showed that in all the cases the best results were achieved using only one hidden layer with 5 neurons. The forecast results for maximum day stock prices were lower than 0.9 (MAPE value) and lower than 2.1 (MAPE value) for minimum day stock prices. Besides, the results indicated that except in one case, in all the cases the results were better using the selection-based inputs instead of all of them.

In (Chae, et al., 2016) an ANN was used for forecasting sub-hourly electricity usage in commercial buildings. Initially, nine potential inputs were considered. In order to measure the importance of each input over the output, the random forest algorithm was used. The results revealed that the inputs related with time (day type and time interval stamp) and the operation condition had the strongest influence. From the rest of the inputs, outdoor

air temperature and relative humidity contributed primarily to energy usage. In fact, the other weather variables were negligible.

For training the net the Levenberg-Marquardt algorithm with the Bayesian regularization was employed. Besides, in each neuron of the hidden layer the logistic sigmoid activation function was selected. To generate the database, every 15 minutes data values were collected from 1 to 31 of July of 2012, weekdays and weekend. However, they had to remove one day data due to the sensor malfunction. Hence, the database was composed of 2880 points. For training, the dataset was divided randomly, while new dataset (August 1-3 of 2012) was collected to evaluate the network.

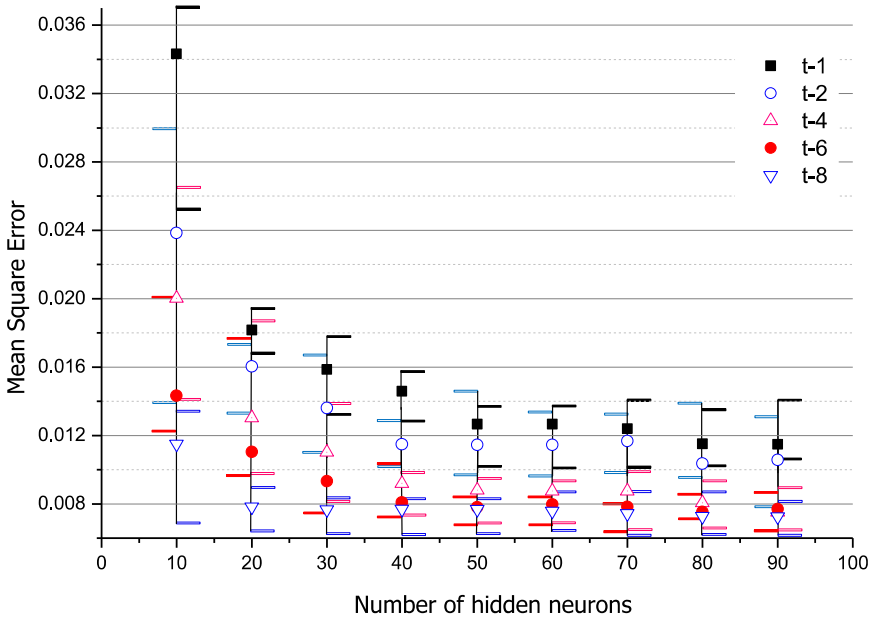


Figure 12 Average MSE in the evaluation with neurons numbers and time delays (Chae, et al., 2016)

To select the network parameters such as neurons in the hidden layer and time-delayed inputs, a comparative study was accomplished. The total number of neurons in the hidden layer varied from 10 to 90, while the delays tested were 1, 2, 4, 6 and 8 (up to 2 hours). Each network configuration was trained 50 times to evaluate the stability and robustness of the model. As the evaluation test shown (Figure 12), the addition of neurons in the hidden layer and time delays could improve the performance and generalization of the net. However, considering the complexity and computation time of the model, the author thought that was reasonable to have 50 neurons in the hidden layer and *t*-6 time delay for input-feedback for the network in their study. For this configuration, the MSE ranges from 0.0071 to 0.0096. It is important to note that in the evaluation procedure the

calculated electricity usage is used as input-feedback instead of the actual ones used in training. Besides, they also studied the influence of the size of training dataset over the network performance. They compared 1, 2, 3 and 4 weeks dataset. The results clearly shown that the best results were achieved with the largest dataset (4 weeks), being the RMSE value 7.3%.

Therefore, it can be seen that feedforward neural networks are widely used for modelling dynamic evolutions. Sometimes they are used for forecasting one-step ahead using as input one time delayed real output instead the predicted output. However, in other cases they are used with predicted outputs as delayed inputs for multi-step ahead prediction. In fact, for multi-step ahead prediction, the predicted output in time *t*-1 has to be the input of the net in the next time-step. Nevertheless, the designer has to manage the input buffer; in other words, the network is not capable to do this and, consequently, it has to be managed externally. It is because of this that some researchers use the recurrent neural networks since it is not necessary to deal externally with the delayed data (i.e. this is done internally).

3.2.2 Modelling dynamic evolutions using recurrent neural networks

Unlike the case of feedforward neural networks, in order to model dynamic evolutions with Recurrent Neural Networks three different approaches can be found (Figure 13). Thus, in most of the cases the aim is to forecast future values (one step ahead or multi step ahead) using for training the net a dataset compound of historical past values (Figure 13 approach 1). Actually, there is available a huge continuous historical data. In this case, the user hast to select the time range (minutes, days, months or years) to train and validate the model. For example, stock exchange, weather, unemployment rate or floods forecasting can be solved with this approach. The next approach is quite similar, however, the training dataset is compound of initial real values instead of historical past values (Figure 13 approach 2). In fact, in this application field, there are not historical values because the dynamic evolutions have an initial point (and not a continuous historical time) such as gear life-cycle where the gear at the beginning is new and with the time degrades. Finally, the third approach is based on the prediction of a complete dynamic evolution without measuring any past or initial value (approach 3). Mostly, the historical or initial values are not available because it is difficult to measure them. Additionally, in other cases the aim is to generalize to new conditions without measuring any data. Thus,

another dynamic evolution is used to train the net and then, the net is able to predict a new complete one without measuring any initial data.

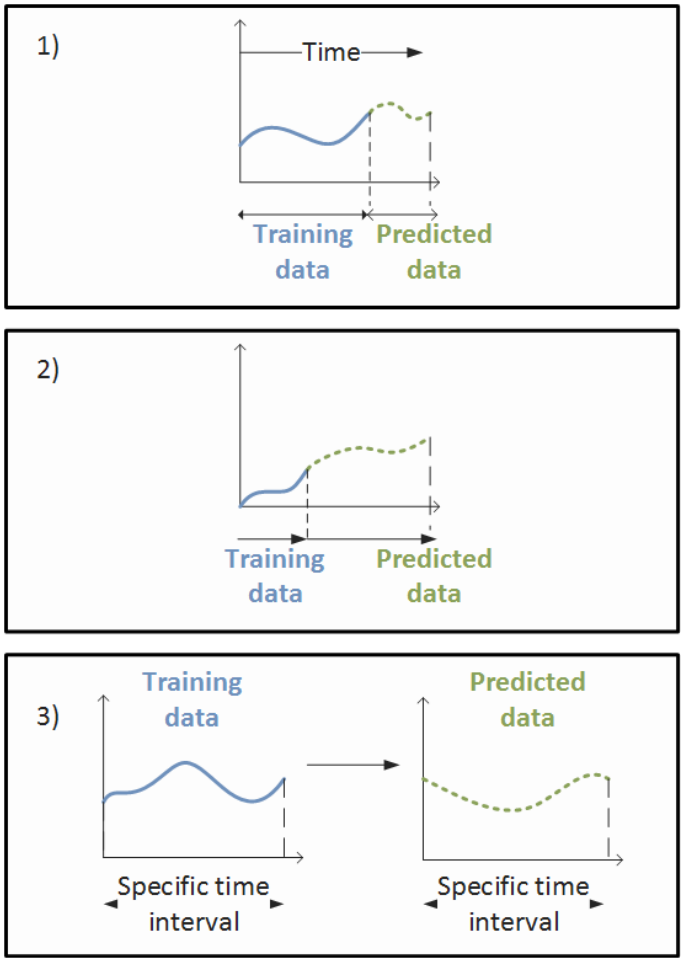


Figure 13 Three different approaches for modelling dynamic evolutions with recurrent neural networks

3.2.2.1 Modelling based on historical data

It can be noticed that most of the application fields analysed in this section make use of historical continuous data, data that do not have a fixed initial point neither a fixed final point. Further, this data is defined by an absolute time, being this seconds, minutes, hours, days, months or years.

In (Liu, et al., 2015) the Elman recurrent neural network was used for predicting the dynamic evolution of the wind speed because, as the author said, due to the local feedback, the network has excellent nonlinear processing capacity.

In this work five wind speed time-series of 800 samples were used. However, before training the network, in the proposed approach, the collected data was pre-processed. First, the original wind speed time series is decomposed into two components (the

appropriate components and the detailed components) using the Wavelet Packet Decomposition (WPD) mathematical algorithm. The detailed components are further decompose into Intrinsic Mode Functions (IFMs) time series adopting the Fast Ensemble Empirical Mode Decomposition (FEEMD) to decrease more the non-stationary components.

Once the data is pre-processed, only one time-series is used to configure the Elman network: the training algorithm and the number of neurons. Thus, the wind speed time series chosen was divided into two datasets, the first 600 points were used to build the prediction model, and the samples from 601 to 800 were used to verify the model. However, in the paper it is not mentioned if the training dataset is divided into training and validation or not, not even what strategy is used to improve the generalization. First, a comparison of different training algorithms was carried out. The results showed that the algorithm that yielded the lowest mean absolute error (MAE), mean absolute percentage error (MAPE) and root-mean-square error (RMSE) was the Levenberg-Marquardt algorithm. However, it is not clear the configuration used to compare the different training algorithms. Based on the figure showed in their work, one could think that 24 neurons in the hidden layer and 1 delay unit in the feedback were used. However, in the paper it is not clear. After selecting the training algorithm, a time series theory based approach was used to select the best number of neurons for the network. Thus, after several computational steps, the best time series was decided as ARIMA (5, 1, 0). Therefore, based on the ARIMA equation, the number of neurons in the input (six) and output layer (one) for the Elman network were selected.

With the network configured, the approach with a WPD-Elman model (with and without the FEEMD pre-processing), Elman, MLP and ARIMA model were compared, with the raw data for the Elman, MLP and ARIMA approaches (for the WPD-Elman de raw data is pre-processed) for one, two and three steps ahead. The results showed that for all the cases the WPD-FEEMD-Elman solution outperforms the other ones.

The Nonlinear Autoregressive with eXogenus inputs (NARX) neural network was used in another work for modelling the photovoltaic (PV) system power production (Vaz, et al., 2016). It is important to highlight that the NARX has a feedback from the output layer to the input, like the Jordan network and, also, tapped-delay inputs (Figure 14).

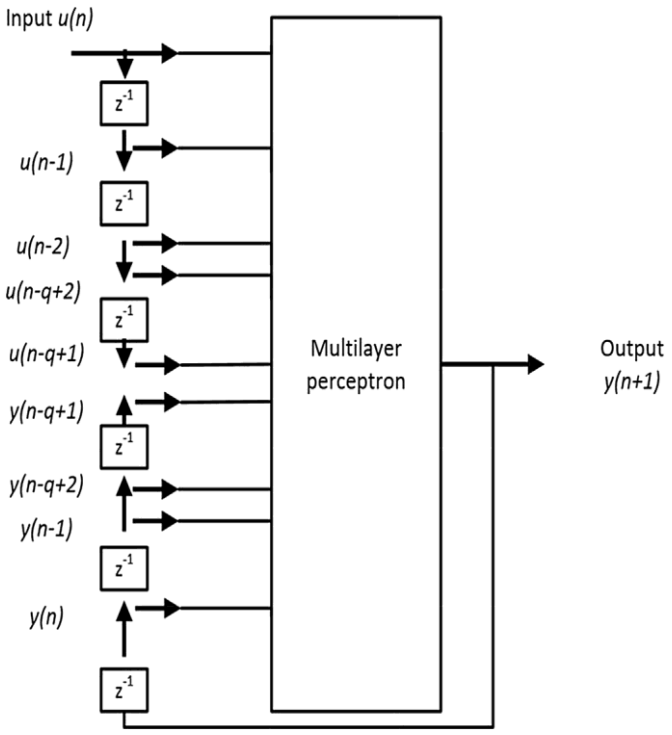


Figure 14 Nonlinear Autoregressive with eXogenus inputs (NARX) neural network (Vaz, et al., 2016)

The inputs selected to model the PV dynamic evolution were time series of meteorological data (solar radiation and ambient temperature) and five geographically separated households PV systems data. These data was collected from 1 August 2012 to 31 July 2013. The observation between 21:00 to 6:30 were removed. Finally, to complete the data pre-processing, all time series were normalized between 0 and 1. Besides, the dataset was divided into two different subsets for different seasons ("winter" and "summer"). In "winter", the data from August 2012 to December 2012 were used to train the net, and the data from January 2013 for forecasting. Similarly, the "summer" dataset was generated using records from February 2013 to June 2013 for training, and records of July 2013 for prediction.

Once the data is pre-processed, it is time to configure the artificial neural network, the NARX network in this case. The authors selected the Matlab™ Neural Network Toolbox's default value, 10 neurons in the hidden layer, because, as they stated, after testing a different number of neurons in the hidden layer the final results were negligible. In the same way, the variation of the number of delays in the tapped-delay-lines (TDLs) in the final results was irrelevant, thus, two delays were used. However, in the paper is not specified if the two delays correspond to the input delays and/or the feedback delays.

Besides, the hyperbolic tangent was selected as the activation function of the neurons in the hidden layer and the linear function for the output neuron. Therefore, due to the hyperbolic tangent activation the data had to be normalized, again, to the [-1, 1] range. Thus, the data was two times normalized. However, no claim was made about the improvements of this decision.

To train the net the Levenberg-Marquardt algorithm was selected and the early stopping method was used to avoid the overfitting and improve the generalization. The training dataset was divided as follows: 70% for training, 15% for validation and 15% for testing. To select the best network, the authors designed five different scenarios to analyse the impact of using variations of the exogenous inputs:

1. Four PV system data as inputs
2. Two PV system data as inputs
3. Meteorological data as inputs
4. Four PV system and Meteorological data as inputs
5. Multistep ahead forecasting

In the first four scenarios, the results shown that the best performance was obtained by combining all the available inputs (4^{th} case). Besides, "winter" forecasts showed lower normalized root mean square error (nRMSE) than "summer" ones. The authors suggest that it might be a consequence of the low variability depicted in overcast days. On the other hand, the difference between one step ahead and multistep ahead prediction was very significant. Indeed, as the authors said, their NARX model appears to be more suitable for short-time prediction horizons.

Also the NARX network was used for predicting oil price movements (Godarzi, et al., 2014). For modelling the dynamic evolution of the oil price eight demand-side factors with potential impacts on oil price movements were analysed. To perform this analysis, first, a time series model was developed. Then, this model was optimized identifying and excluding the negligible inputs using a step-by-step t-statistic analysis. Finally, using Interdependent Variable Lags (IVL) the most important variable lags over the output were identified. Thus, for some inputs the actual value was selected while in other inputs the value t-1 or t-2 were chosen.

Once the time series model was defined, the NARX model inputs and lags were chosen based on the time series model. The data from 1974 to 2004 were used to train the net,

and the data from 2005 to 2009 were used to test the NARX model. Besides, the input and output data were normalized within the range [-10, 10] and [-1, 1], respectively, in order to obtain more precise results while reducing the required computing as authors alleged. On the other hand, the optimum number of neurons in the hidden layer was found using by trial and error, being this number 25.

The results exposed that the prediction of the net was accurate (Figure 15). Besides, they compared the performance of the NARX net with a feedforward neural network and the time series model. The mean absolute error (MAE) shown that the NARX model (4.96%) accuracy was clearly higher than the time series model (6.47%) and the feedforward neural network (8%).

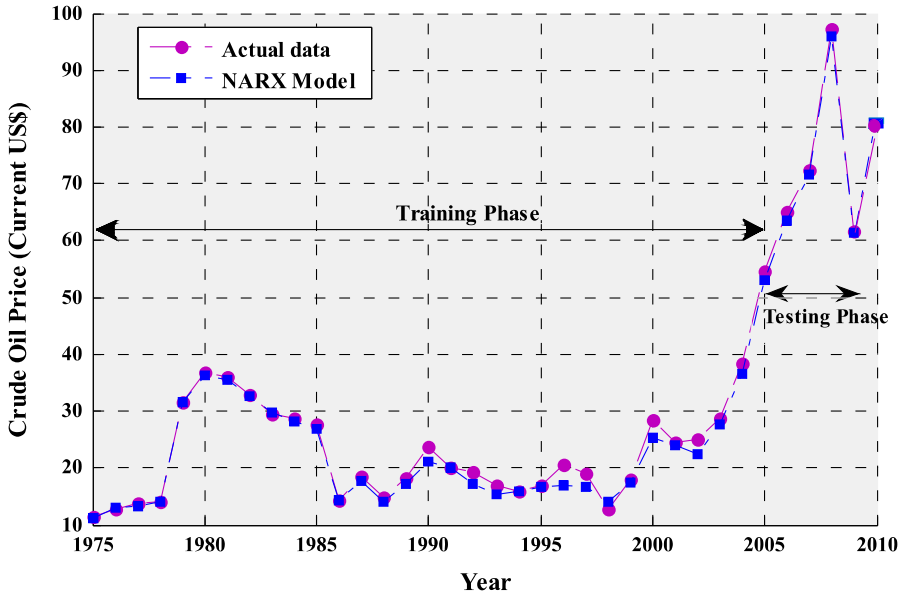


Figure 15 Comparison of the NARX model predicted oil price versus the real one (Godarzi, et al., 2014)

In another work, a recurrent neural network was used for predicting the urban stormwater runoff (Zhang, 2011). The input data used were precipitations and discharges from 31 July to 20 November 2010. The raw data were randomly divided into 34721 time steps. Thus, 70% of the data were used to train the network, 15% for validation, and the rest 15% for testing the network.

The RNN architecture with lagged inputs and feedback from the output layer to the input layer was selected. Besides, the activation function of the neurons in the hidden layer was the sigmoid one, while for the output neurons the linear activation function was chosen. The network was trained using the Levenberg-Marquardt backpropagation algorithm. In

addition, the early stopping was used to avoid the overfitting of the net during the train and improve the generalization of the net. It is significant to remark that the training of the net was performed in open loop. Open loop training allows to train the net using the real past values to predict the correct ones. After training, the net may be converted to close loop.

Finally, the number of neurons in the hidden layer and the number of delays were selected by trial and error after training the net plenty of times. The best number of neurons was 50, and the best number of delays in the tapped delay was 11. These numbers were the maximum affordable not to be out of memory. In the paper it was not said if the delays correspond to the input layer, feedback or both. With this configuration the correlation between the output and the target felt approximately the 95% confidence. Thus, the conclusion was that the model was adequate for modelling the urban storm runoff.

Also, another Elman modification known as the Layer-Recurrent Neural Network (LRNN) was used for 5-day wind and power forecasting. This network, as the Elman, has a feedback from the output of the hidden layer to the input. However, the LRNN provides more flexibility because it is possible to have more than one hidden layer and different transfer functions in each layer (Beale, et al., 2012).

In this particular case the objective was a bit different from the previous works. In fact, it could be said that it is a bridge between modelling dynamic evolutions using historical values and modelling evolutions using initial real (measured) values. In particular, one-month time series data of six parameters sampled with 5 and 10 minutes time steps were obtained from two different wind stations A and B, respectively. Therefore, in each station 22.320 points were collected. However, the difference is that the data were used as follows: the first day of the month was used for training the net, and the next 5 days were predicted by the net. Then, the first 6 days real data were used for training and the rest 5 days were predicted and so on. Thus, the training was done using the real data instead of the predicted one.

Therefore, two independent networks were used for modelling the dynamic evolution of wind speed and wind power. Each net had five inputs and one output. Besides, only one feedback delay was selected without carrying out any analysis. However, for the neurons in the hidden layer a comparison was done for three different numbers of neurons: 5, 15 and 24. The Levenberg-Marquardt algorithm and early stopping were selected for training and avoid overfitting, respectively. Besides, for the hidden layer neurons the hyperbolic

tangent transfer function was used while in the output neurons the linear activation function was used. However, it is not mentioned in the paper if the data were normalized or not, neither how many times each network configuration was trained.

The results shown that the LRNN with five neurons in the hidden layer was almost useless. However, in the station A the best results were yielded with 24 neurons in the hidden layer whereas in the station B the lowest errors were produced with 15 neurons in the hidden layer. Besides, new data was selected and a comparison was made between the results achieved with the station A (with 24 neurons in the hidden layer) and data collected from a new station, not used in the previous analysis. In both cases the results were quiet similar. At station A an overall root symmetric mean absolute percentage error (sMAPE) value of 0.003%, mean absolute scaled error (MASE) value of 0.46%, and standard error of 0.968% were yielded, while in the new station the error values were sMAPE of 0.003%, MASE of 0.676%, and standard error of 1.129%. However, it must be noted that for the new data the network was retrained, i.e. it is not used the network trained with the data obtained from station A.

3.2.2.2 Modelling a complete dynamic evolution with initial measured values

This strategy is quiet similar to modelling dynamic evolutions using historical real values, mostly one-step ahead prediction is used for modelling the dynamic evolution. Unlike modelling with historical values, the prediction window is bounded, i.e. it has an initial point and an end point, hence, there is not an historical continuous past data available.

In order to model the dynamic evolution of the health condition of gears a modified RNN was used (Tian & Zuo, 2010). In particular, the modified RNN had a feedback from the output layer to the input (Jordan network) and another feedback from the output of the hidden layer to the input (Elman network). However, there was a slight difference. In the Jordan context layer (feedback layer) one neuron obtains the predicted output one-time delayed whilst another neuron obtains the error between the real and predicted output one-time delayed, thus, both the one-step delayed output and the error are feedbacked to the input of the network. Therefore, this network architecture only can be used for one-step forecasting because the real output value at t-1 is necessary. Besides, two neurons in the input layer were used because, as the authors stated, every data point in a time-series is only strongly dependant on the previous two values. Finally, the number of neurons in the Elman context layer was equal to that in the hidden layer. However, no mention was made about the feedback delay or self feedback weight.

The total duration of the data used was 18.8 hours and 141 vibration data points. In fact, at the end of the 18.8 hours a gearbox failure occurred. However, not all the data were used because up to 110 data points the RMS value was rather flat. Thus, only data points from 90 to 141 were used because the objective was to predict the health condition of the gearbox when it starts to deteriorate. Actually, data points from 90 to 135 were used to train the net and the point number 136 was predicted. Then, the network was trained using data points 90 to 136 and the point number 137 was predicted and so on.

For training the net, the exact gradient based training algorithm was selected. Besides, two neurons in the hidden layer were chosen based on the size of the problem, as it was said in the article. In the neurons of the hidden layer the sigmoid function was used and the linear one in the output neuron. However, no reference is done about the data normalization. Finally, the training and prediction process was conducted ten times, and the average of the normalized mean squared error (NMSE) was used to compare the prediction performance.

The average NMSE for training and prediction were 0.0012 and 0.1183, respectively (Figure 16). Besides, a comparative study was carried out to compare the results of the proposed RNN architecture and another RNN architecture, the fully connected recurrent neural network (FCRNN). Furthermore, for each architecture two different configurations were used: for the ERNN networks, one and two neurons in the hidden layer were used, respectively, while for the FCRNN two and three were selected, respectively. The comparison results showed that the ERNN outperforms the FCRNN network in both cases, i.e. one and two neurons in the hidden layer. Besides, the ERNN network with two neurons in the hidden layer yielded lower NMSE error than the network with one neuron in the hidden layer.

It can be noticed that the strategy is quiet similar to modelling dynamic evolutions using historical real values, and mostly one-step ahead prediction is used for modelling the dynamic evolution. Unlike modelling with historical values, the prediction window is bounded, i.e. it has an initial point and an end point. However, as described in the next section, other works have tried to predict the complete dynamic evolution bounded in time, without using any initial value for training the net. Moreover, other dynamic evolutions are used for predicting new ones.

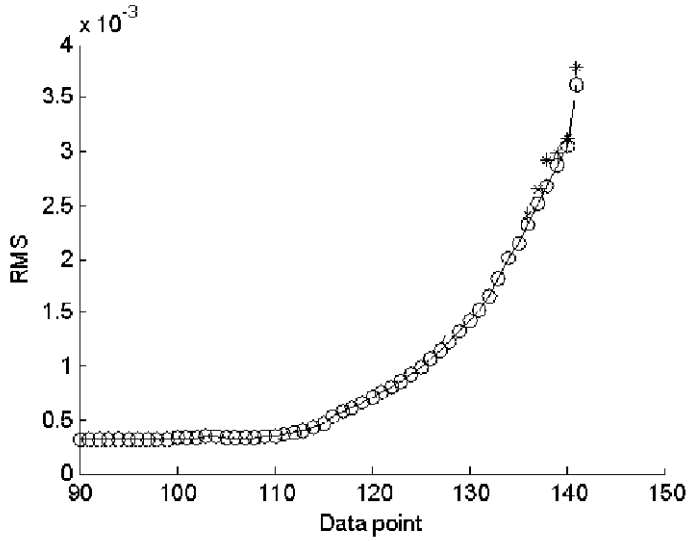


Figure 16 The prediction results. (o) represents the real value and the (*) the predicted value (Tian & Zuo, 2010)

3.2.2.3 Modelling based on another complete dynamic evolution

When it is not possible to measure the historical or initial data, a new and complete dynamic evolution has to be predicted. For that, the net is trained using a complete dynamic evolution and, after, the net is capable to predict a new one.

In (Teufel, et al., 2003), the objective was to model the glucose metabolism using the Elman network. Thus, the model was able to predict the glucose for three days without measuring the glucose of previous days neither at the beginning of the first day. However, as suitable patient data were not available, an analytical glucose model was used to generate a training and validating dataset. Thus, the train and validation dataset was generated for three days with a temporal resolution of one minute.

The network inputs were the ingested carbohydrates and the injected insulin. Actually, these inputs were the same inputs as the ones considered in the analytical glucose model. Both inputs were events that occurred only a few times a day. Besides, only one output neuron with linear activation function was used for blood glucose concentration. On the other hand, for the neurons in the hidden layer the hyperbolic tangent function was selected. These parameters were fixed before the training process. However, the neurons in the hidden layer varied from one to ten. Besides, during the training the weights including the self feedback weights, which make reference to the data to remember in the feedback, were adjusted.

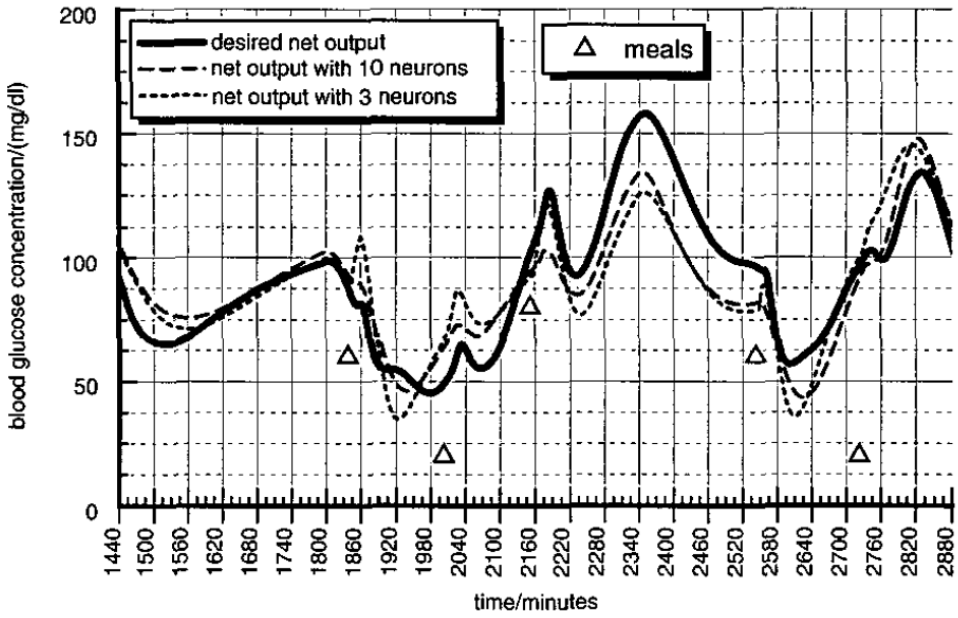


Figure 17 Training results of the second day (Teufel, et al., 2003)

The training results showed that the ten-neuron net yielded the best results for the training data (Figure 17). Nonetheless, no reference was done to the best self feedback weight. For testing the generalization capabilities of the trained net, the amounts and times of meals and insulin injections were varied. The results clearly indicated that the net with more neurons in the hidden layer modelled better the glucose metabolism (Figure 18).

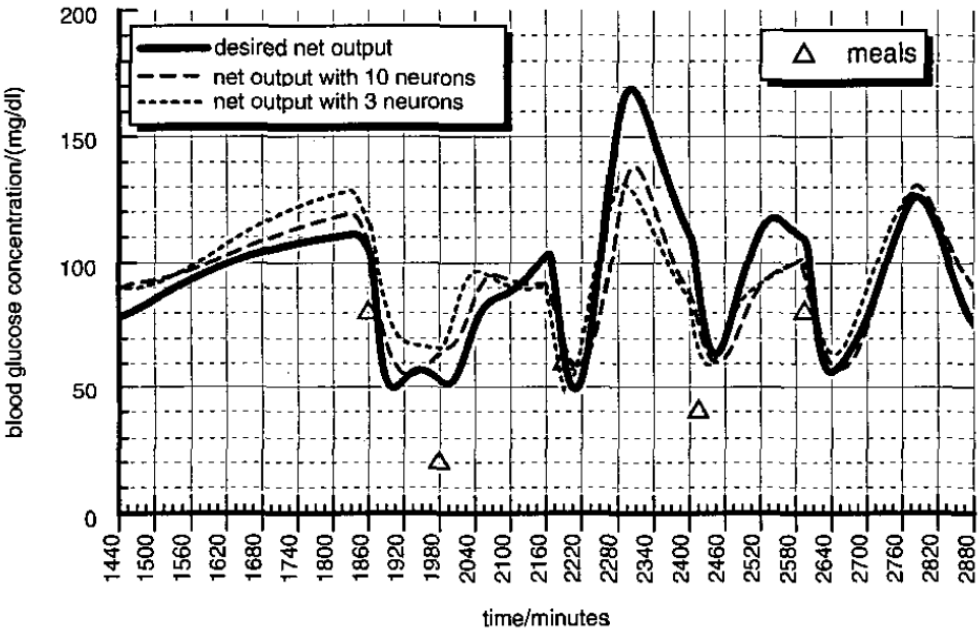


Figure 18 Generalization test of the second day (Teufel, et al., 2003)

3.3 Conclusions

In real life, most of the systems have a dynamic behaviour and actual events that depend on past events. When dealing with this type of systems, ANNs have shown that are suitable and widely used for modelling those dynamic evolutions. In order to obtain the best reliable models, the data selection and pre-processing are vital to achieve the best prediction performance. In some cases, it is possible to select the best inputs using different types of analysis such as regression or ARIMA, whereas in other cases the knowledge of the experts in the topic to model is crucial.

The review of the state-of-the-art shows that in most of the cases the prediction of dynamic evolutions is based on the past or historical events. Thus, the ANN is trained with past historical data for predicting future events. Mostly, these predictions are one-step ahead, ergo, for predicting $t+1$ event, t real (measured) values are used. For the one-step prediction, feedforward neural networks or recurrent neural networks are used. In fact, although the RNN are more powerful, in one-step forecasting the feedforward neural networks have shown great performance. This is because in one-step forecasting the past real values lagged inputs are highly important and the feedforward neural networks can use them with few external manage by the designer.

However, for multi-step ahead the problem is more complex. In this case, for predicting future events the network is not fed with real (measured) past values (Section 3.2). Actually, in t+1 the output is usually predicted using the t, t-1, … t-n output predicted values depending of the prediction horizon i.e. usually, if the aim is to predict three-step ahead, t and $t-1$ are predicted values and $t-2$ is a real value. It is true that it can be accomplished with feedforward neural networks. However, the designer has to develop an external program to convert the predicted outputs into network inputs for the next time step. For recurrent neural networks this explicit management is not necessary because the net performs the data transfer by itself through its feedbacks. Thus, it is said that the RNNs have "memory" thanks to the feedbacks. The review of the state-of-the-art shows that three main RNN topologies are used for modelling dynamic evolutions using historical values: the Elman network, the Jordan network and the NARX. However, it is possible to develop a new and more specific RNN using new or combining different feedbacks.

Aside from modelling dynamic evolutions using historical past values, in other cases, the objective is to predict the future events using for training initial values. The training strategy is almost the same. Moreover, in this application field also one-step or multi-step

ahead strategies can be used. Thus, the only difference is based on the application field. Actually, in these cases the prediction window is fixed. It has an initial start point and an end point.

However, some application fields require a third approach that is totally different. In this approach, historical or initial data are not available, often because measuring real values is time and resource consuming. In those cases, the aim is to predict a complete dynamic evolution but without initial or real values. Thus, the training strategy is completely different. First, the inputs are different from the output of the network. Second, the prediction cannot describe neither one-step ahead neither multi-step ahead because a new complete dynamic evolution is predicted. Finally, it makes no sense to use feedforward neural networks because the network should have as many inputs as time-steps the complete dynamic evolution has. Therefore, it is highly recommended, if not mandatory, to use recurrent neural networks.

Another important aspect to highlight is the generalization: given the nature of the undertaken applications, the scope of generalization in the literature review is limited to the target dynamic evolution, i.e. usually only one dynamic evolution is used for training the net and another one is predicted.

Therefore, in order to model the specific grinding energy, wheel wear and surface roughness, it is crucial to predict a complete dynamic evolution without measuring any initial real value. Besides, as said in section 2.3, it is crucial to break with the wheel-workpiece pair in order to develop more industrial useful models. Thus, a variant of the third dynamic evolution modelling approach is proposed with a higher generalization capability.

Besides, it is reasonable to think that one ANN will not be capable to model all the grinding wheel and grinding conditions. Thus, under the selected application field, a methodology for generating new ANNs for different application fields should be developed with generalization capabilities concerning new grinding wheels and new grinding conditions.

4 ANN BASED STRATEGY FOR ESTIMATING GRINDING DYNAMIC EVOLUTION VARIABLES

The ANN strategy for estimating grinding dynamic variables can be divided into two main sections. On one hand, the first section is the configuration founded on the knowledge of the process. Hence, the ANN input/output and architecture are decisions taken through the analytical models and application objectives. However, other decisions like the training algorithm or the generalization method are taken after studying the state of the art. The ANN inputs are selected based on the analytical models and the Layer-Recurrent Neural network is chosen since it is more powerful for modelling dynamic evolutions and because it has a feedback from the output of the hidden layer to the input of the hidden layer, thus, it functions without any initial output value. As to the ANN training configuration, the Levenberg-Marquardt training algorithm is selected due to the fast convergence. Besides, for improving generalization, the Bayesian regularization is used. Finally, in order to select the best network, a two-phase methodology is presented. First, the phase called "coarse tuning" is done. During this phase the aim is to obtain the network structure around which the lowest MSE test errors are yielded. Given the best structures inferred in the coarse tuning, the so called "fine tuning" is carried out in neurons in the hidden layer-delays structure taken one by one within the specified range. During the fine tuning, the MAME metric is used instead of the widely used for evolution analysis Mean Absolute Percentage Error (MAPE) because from the grinding process point of view is easier to measure the error using absolute errors instead of percentage ones.

In order to model the dynamic evolution of the grinding main variables such as wheel wear, surface roughness and specific grinding energy a general methodology is described in this chapter. This methodology is based on the knowledge of the grinding process.

Thus, first, the general strategy is described. This strategy is based on the analytical models that link different grinding variables. Second, based on the general strategy, the ANN based methodology is described. This description is based on selections such as the ANN inputs and outputs or architecture. In fact, these decisions are based on the knowledge of the process extracted from the analytical models.

4.1 General strategy

The objective is to model the complete dynamic evolution of the surface roughness, the wheel wear and the specific grinding energy. Thus, the equations 1 and 2 (Section 2.2.1) show that the surface roughness and the specific grinding energy are closely related with the wheel wear. Besides, these variables are also related with the wheel characteristics and cutting conditions. However, this relationship is highly non-linear. Therefore, the aim is to model this non-linear relationship using artificial neural networks.

As said before, the surface roughness and the specific grinding energy are related with the wheel wear, expressed through the product $C{\cdot}r$, where C is grain density and r is the grit shape factor. Commonly, $C{\cdot}r$ can be considered as a single factor related to the surface topography and surface wear of the grinding wheel. Therefore, the surface roughness and the specific grinding energy are dependent on the state of the wheel wear. Actually, these two variables are not static values and change while more parts are machined (see Figure 5 of Section 2.1). Thus, for a more reliable model of the surface roughness, the wheel wear and the specific grinding energy, it is highly important to model the dynamic evolution of them while the state of the wheel is changing. In addition, the models have to show the relationship between the wheel characteristics and the cutting conditions. Thus, in order to show this relationship, the inputs of the model have to represent both the wheel characteristics and cutting conditions.

On the other hand, as show in Section 3, the ANNs are suitable for modelling highly nonlinear systems like those represented by the equations 1 and 2 of Section 2.2.1. Besides, artificial neural networks are excellent solution for modelling dynamic evolutions (Section 3.2). Actually, one usual application of ANNs is modelling dynamic evolutions where the future events are predicted based on past or previous events. However, in this case there are not data available because the aim is to model the grinding variables without measuring or even carrying out the operation. Thus, the strategy for

ANN modelling of the dynamic evolution of grinding variables is focused on giving (predicting) the real value of the complete dynamic without initial real values with the capability to generalize to new grinding wheels and grinding conditions not used during the training of the net.

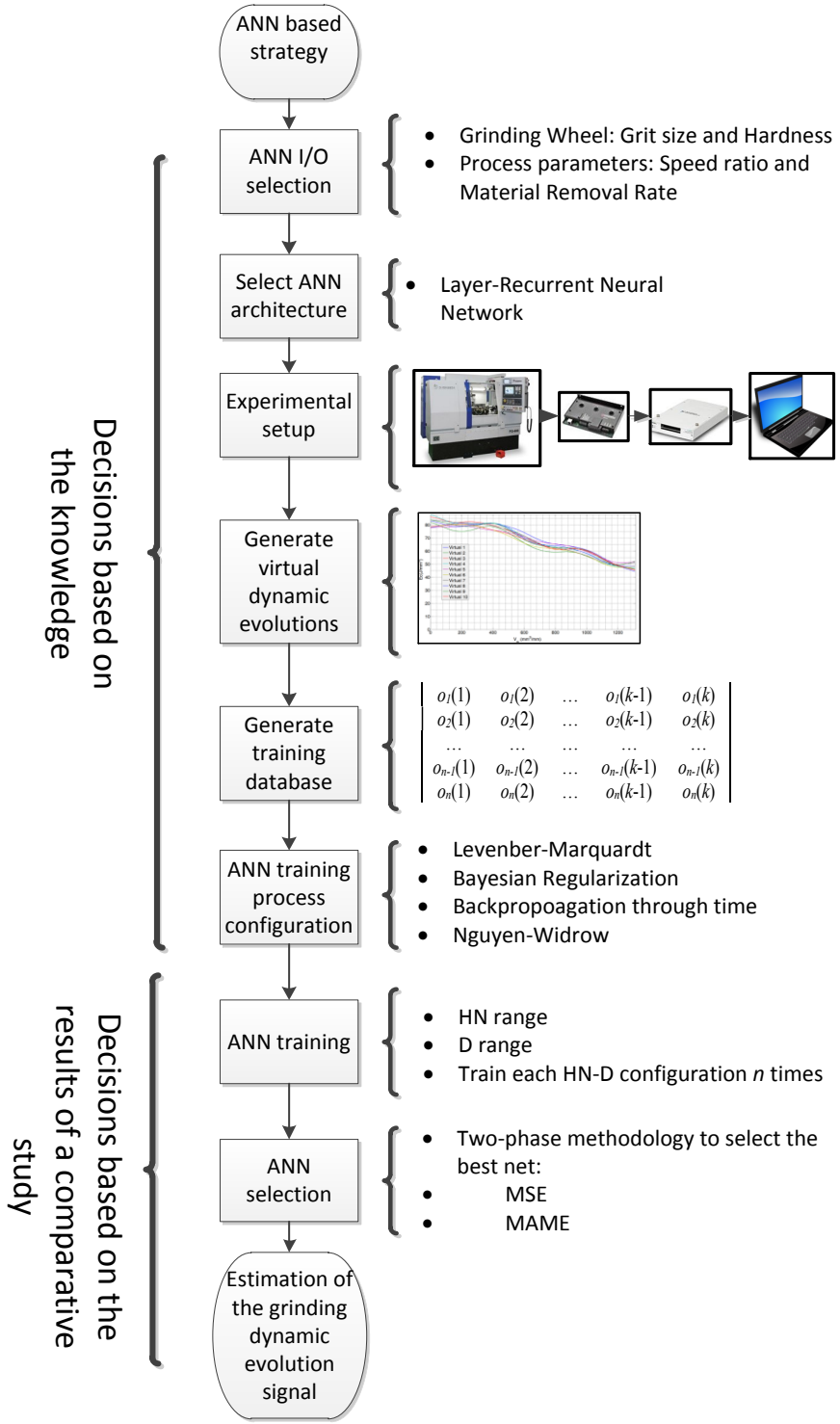


Figure 19 Diagram of the ANN based strategy for estimating grinding dynamic evolution signals

Therefore, as can be seen in Figure 19, the ANN strategy for estimating grinding dynamic variables can be divided into two main sections. On one hand, the first section is the configuration founded on the knowledge of the process. Hence, the ANN input/output and architecture are decisions taken through the analytical models and application objectives, as well as restrictions such as generalization to new grinding wheels or predicting complete dynamic evolutions without initial real values. However, other decisions like the training algorithm or the regularization method are taken after studying the state of the art.

On the other hand, the second section refers to decisions based on the results of a comparative study. For example, the network is trained with different number of neurons in the hidden layer and comparing the training results the appropriate number of neurons in the hidden layer is chosen. In fact, in most of the works related to Artificial Neural Networks, parameters such as the number of neurons in the hidden layer are selected with a trial-and-error approach.

Finally, it is noteworthy that the initial aim of this work is not to create new architectures, training algorithms or new generalization techniques. Moreover, one of the pillars of the work is to use existing architectures, training algorithms and generalization techniques that can be found in commercial software packages. For this work, the Neural Network Toolbox provided by Matlab^{TM} has been used.

4.2 ANN configuration

When addressing the use of ANNs, some considerations must be taken related with the inputs and outputs of the network and the network architecture.

Besides, it seems unlikely that one unique ANN could be enough for modelling the whole problem space since all the grinding wheels commercially available and grinding operations cover an extremely large number of components and requirements. Since one of the objectives of this work is to show the potential of the methodology for modelling the dynamic evolution of the specific grinding energy with RNNs, an average area of application related to the grinding of steel parts with non-extremely demanding surface finish has been selected.

4.2.1 ANN input/output selection

Selecting the inputs and outputs of the Artificial Neural Network is vital because the performance of the final network is heavily dependent on the input variables used for developing the ANN. Thus, in this work the inputs selection criteria are based on the analytical model available and the knowledge of the researchers in the field of grinding process.

Besides, based on the mathematical expressions of equation 1 and 2 (Section 2.2.1), grinding wheel characteristics and grinding conditions are used as inputs of the network. On one hand, for the application field selected, the grinding wheel is characterized by the grit size and the wheel hardness. These inputs are correlated with the product $C{\cdot}r$. In Section 2.1 a brief description of these inputs is done. On the other hand, for characterizing the grinding process, the specific material removal rate (Q') and the speed ratio (q_s) are used. The specific material removal rate (Q') is defined as the material removal rate of the workpiece per unit width of wheel contact while the speed ratio (q_s) is the ratio of the wheel speed over the workpiece speed (Marinescu, et al., 2004). Actually, these inputs describe the operation carried out and are previously adjusted in the machine Computer Numerical Control (CNC).

4.2.2 ANN architecture

The architecture refers to how the artificial neurons are connected between them. As explained in Section 3.1.3, one artificial neuron is not enough to model the complex relationship between the inputs and the outputs of the system, then, frequently more than one neuron have to be used, i.e., more than one artificial neuron working in parallel are needed. Each of this parallel working neurons set is denoted as a 'layer'. Therefore, combining one or more layers' new network architectures are designed to solve a specific problem. As shown in Section 3.2, the researchers have made used of feedforward or recurrent neural networks for modelling dynamic evolutions. Multilayer perceptron neural networks with lagged inputs have shown good performance for modelling dynamic evolutions one-step ahead ($t+1$). Moreover, with out-network manage it is possible to use feedforward networks for multi-step ahead prediction.

However, in the most restrictive case (the estimation of the specific grinding energy) where all the network inputs (wheel characteristics and grinding conditions) are constant values it is impossible to use feedfordward networks because for each "time-step" of one

complete evolution the inputs have the same value. However, RNNs have a feedback of the output of the network and/or the hidden layer and this feedback gives information about the previous states. Thus, the network has a "new input" about the previous states of the output of the network and/or the hidden layer. Besides, the RNN have shown better performance dealing with dynamic evolutions due to the feedbacks of their network structure. In fact, it is said that these feedbacks give to the network "memory" capacity.

Amongst the recurrent neural networks used for modelling dynamic evolutions, the Jordan network, the Elman network and the Nonlinear AutoRegressive network with eXogenous inputs (NARX) are the most used ones. When using the RNN for multi-step prediction, the use of recurrent neural networks with output feedback (Jordan and NARX) and hidden layer output feedback (Elman) used to be quite different, though.

In the case of Jordan and NARX networks, mostly, the network is used with real output values, and the real output values of the feedback are gradually replaced by the predicted ones. Hence, at the first steps the network is used with the real values and when the prediction is going forward the real values are replaced with the predicted outputs. Thus, after that point the network works feed backing only predicted output values.

However, with the Elman network is not possible to work like with Jordan or NARX networks because the output values of the output layer are not available. Therefore, the network handles to fill the feedback and provide feedback inputs to the net. Besides, the literature review has shown that the unique case where the aim was to predict the complete dynamic evolution without any initial or past value the Elman network was used with great results.

Therefore, in this work the Elman based Layer-Recurrent Neural Network (LRNN) provided by the Neural Network Toolbox (Matlab^{TM}) is used to model the dynamic evolution of the wheel wear and surface roughness in order to develop a reliable smart sensor as well as an estimator of specific grinding energy. The LRNN has a feedback from the output of the hidden layer to the input of the hidden layer, thus, it functions without any initial output value. However, unlike the Elman network, with the LRNN it is possible to have more than one hidden layer and different transfer functions in each layer.

Therefore, the selected network architecture has at least one hidden layer and a feedback from the output of the hidden layer to the input of the hidden layer. However, in this work only one hidden layer is used because a single hidden layer with enough sigmoid neurons and linear output neurons can approximate any continuous function (Cybenko, 1989). Thus, the selected network is similar to the network in Figure 10.

4.2.3 Experimental database

Under supervised learning, two factors have direct influence on the performance of the neural network: First, the ANN architecture and structure, and second, the representational accuracy of the training dataset (Philip, 2009). Thus, in order to get a suitable training and testing dataset, it is crucial to design a proper methodology of experiments that adequately cover the problem to be solved, as well as to configure the required acquisition system, together with the proper measuring devices and procedures, so as to acquire and store the experimental database.

4.2.3.1 Experimental set-up

As stated before, an average area of application related to the grinding of steel parts with non-extremely demanding surface finish has been selected as application field to demonstrate the potential of the presented methodology. Therefore, for this application field, the right grinding wheels and grinding conditions must be selected. The experiments carried out with these grinding wheels and grinding conditions will be used to generate the training dataset.

In particular, the following grinding wheels are used to carry out the experimental tests that compose the training dataset:

- 82AA36K6VW
- 82AA70G6VW
- 82AA100G6VW
- 82AA100J6VW
- 82AA701J6VW
- 82AA36G6VW

In the notation of the grinding wheels (see Figure 20) the first number and letters (4 in total) make reference to the abrasive type (see Section 2.1). In this case, all the grinding wheels use the same alumina abrasive type (82AA). Then, the next two numbers denote

the grit size of the wheel. In particular, the wheels with 36, 70, 100 and 701 grit sizes are used. The next letter makes reference to the hardness, K, G and J in this case. The next number is the reference number of the wheel structure. It can be noticed that in this work all the grinding wheels have the same structure number (6). Finally, the last two letters designate the bond material of the grinding wheel and, again, all the wheels have the same vitrified bond (VW). Thus, only the grit size and the hardness are used as network inputs.

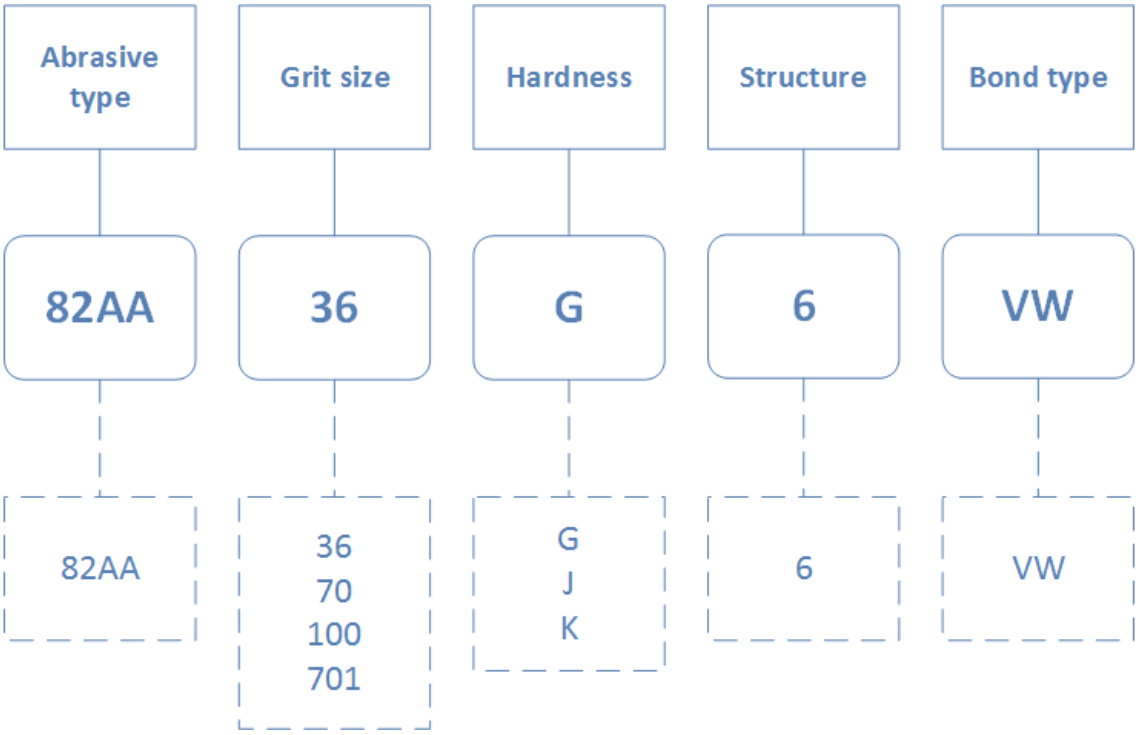


Figure 20 Scheme of the grinding wheels nomenclature used in this work

On the other hand, as said in Section 4.2.1, the specific material removal rate (*Q'*) and the speed ratio (q_s) are used for describing the grinding operations. In the experiments, the following values are used:

- Material removal rate (*Q'*) ($mm^3/mm \cdot s$): 1, 2.5 and 4
- Speed ratio (q_s): 60, 80 and 100

The grinding wheels and operation conditions of the experiments are summarized in Figure 21.

Each experiment involves grinding a total amount of workpiece material up to 40,000 mm^3 because it is dependent on the wear of each wheel. Usually, grinding variables are referred to the unit wheel width (Marinescu, et al., 2004). In this case, since wheel width is 20 mm, the total specific volume of part material removed (V'_w) in each experiment is,

therefore, up to 2000 mm^3/mm. For an industrial example, in which competition motorcycle components are ground to the final shape, this value of V'_w corresponds to an approximate number of 1300 machined parts. Actually, performing these experiments is a highly cost and time consuming task.

Figure 21 Grinding wheels and grinding conditions used to carry out the experiments to generate the training database

4.2.3.2 Acquisition system and measuring devices

In order to train an ANN a database has to be generated. Thus, in order to create this database, experiments are carried out using the grinding wheels and cutting parameters shown in Figure 21.

On one hand, during each experiment, values of surface finish and wheel wear are periodically measured, at different values of part material removed (V'_w). Wheel wear can be calculated by periodically machining an aluminium plate and comparing the difference in wheel diameter. This is a common technique at lab scale (Marinescu, et al., 2004). Likewise, the actual roughness of the machined part is known by using a profilometer. In fact, four times is measured the surface roughness with the profilometer and, then, the mean is calculated. The measured values of both wheel wear and roughness are manually stored in software files known as *experiment files* so as they can be employed in the further training process.

On the other hand, for collecting the specific grinding energy, this can be easily calculated measuring the power consumption at the wheel spindle (see Figure 22). Thus, first the power consumption of the grinding machining is transformed into voltage signal using the Hall-effect based transducer UPC-FR provided by Load Controls. Then, this analog signal is transformed into digital signal using the A/D converter NI USB-6251 of National Instruments. This signal is stored at a sample and storage rate of 100 Hz in a PC. Finally, the acquired power consumption is transformed into specific grinding energy using the following equation:

$$e_c = \frac{P}{Q_w} \tag{9}$$

where P is the power consumption at the wheel spindle and Q_w is the material removal rate of the operation.

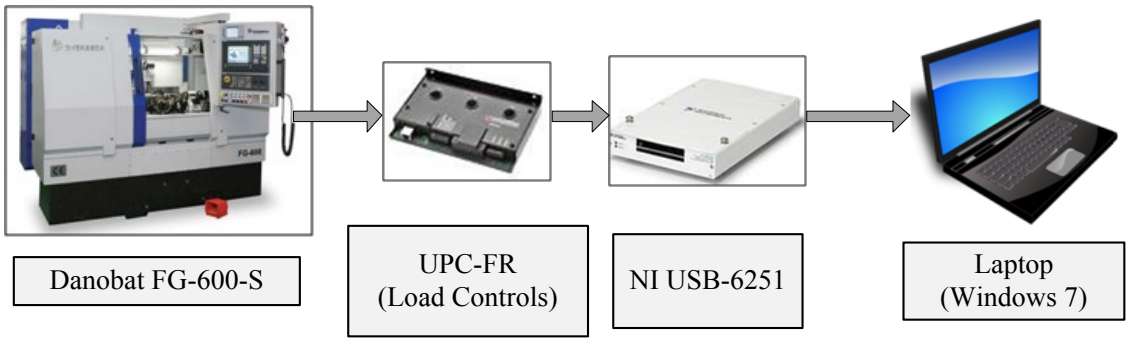


Figure 22 Acquisition system for collecting power samples

4.3 ANN training strategy

One of the main characteristics of the ANNs is their capability to learn from examples. Thus, the network is trained in order to learn the non-linear relationships between the inputs and output of the net. In fact, some network parameters are adjusted with real measured values during the training process. In this case, the parameters to adjust are the network weights, the biases, the number of hidden neurons (HN) and the number of delays (D) of the feedback. For that purpose, the network is trained with real experimental tests available in the previously defined database. Thus, first the training of the network has to be configured and then the training dataset.

4.3.1 ANN training configuration

As said in Section 3, one of the main characteristics of the ANNs is their capability to learn patterns through examples. All the approaches analysed in Section 3.2, use the supervised learning paradigm. Thus, to the given network inputs the desired outputs are showed during the training phase. Since in this case the desired outputs are available, in this work also the supervised learning is used.

4.3.1.1 Training algorithm

Concerning the training algorithm, different approaches can be found. The most common is to use a backpropagation (BP) based training algorithm to minimize the output error and update the network weights and biases. In some cases optimization methods not based on backpropagation such as genetic algorithms or Particle Swarm Optimization (PSO) can be used in order to minimize the output error and found the global minimum for a giving weights and biases. However, when the network size is big (considerable number of weights and biases) the optimization methods not based on backpropagation are slow. Furthermore, with the LRNN the number of weights increases significantly due to the neurons in the context layer due to the feedbacks. Thus, in this work a backpropagation based learning algorithm is used.

Among different backpropagation based training algorithms, one of the most used is the Levenberg-Marquardt optimization algorithm due to its fast convergence and efficiency (Hagan & Menhaj, 1994). While backpropagation is a steepest descent algorithm, the Levenberg-Marquardt algorithm is an approximation to Newton's method. The Levenberg-Marquardt modification to the Gauss-Newton method is:

$$\Delta x = [J^T(x)J(x) + \mu I]^{-1} J^T(x)e(x) \tag{10}$$

where *J(x)* is the Jacobian matrix with the first derivatives of the network errors respect to the network weights and biases, *e* is the network errors vector and μ is a parameter that when is large the algorithm become steepest descent and when is small the algorithm becomes Gauss-Newton.

Likewise, back-propagation through time (BPTT) is applied instead of the usual backpropagation (Werbos, 1990). BPTT is an adaptation of the BP training algorithm for RNNs that is a very powerful tool for dynamic modeling, among others. In fact, it tends to accelerate the training of Recurrent Neural Networks. As explained in Section 3.1.3,

unlike the feedforward neural network, the RNNs have at least one feedback connection. Therefore, the basic idea behind the BPTT is "unfolding" the RNN in order to convert it into a feedforward neural network and, thus, apply the backpropagation training algorithm (see Figure 23).

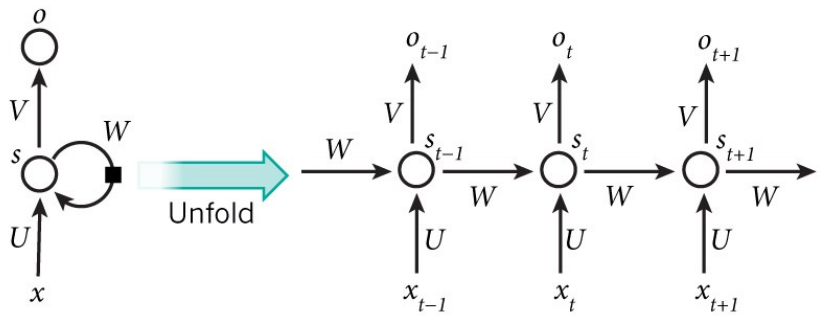


Figure 23 Unfolded RNN (http://www.wildml.com/)

4.3.1.2 Generalization

On the other hand, to improve the generalization capabilities of the network the Bayesian regularization is used. The Bayesian Regularization gives better generalization capabilities compared to early stopping (Dan Foresse & Hagan, 1997). Besides, the Bayesian regularization does not require the use of validation data and, consequently, more data can be used to train the net. As said before, the training objective is to reduce the sum squared error (E_D) of the difference between the target value and the predicted value. Bayesian Regularization adds the sum squared of the network weights (E_W) to the equation (MacKay, 1992):

$$F = \beta E_D + \alpha E_W \tag{11}$$

where F is the objective function and α and β are the objective function parameters. The main problem of the above equation is to find the correct values for α and β. To solve this problem the Bayesian framework was proposed (MacKay, 1992), where the network weights are considered random variables:

$$P(\mathbf{w}|D,\alpha,\beta,M) = \frac{P(D|\alpha,\beta,M)P(\mathbf{w}|\alpha,M)}{P(D|\alpha,\beta,M)} \tag{12}$$

where D represents de data set, M represents the neural network model and $\mathbf{w}$ is the network weights vector. It is assumed that the noise in the training data set is Gaussian

and that the prior distribution for the weights is Gaussian. The optimization of the parameters α and β require solving the Hessian matrix of $F(\mathbf{w})$ at the minimum point, which is possible with the Levenberg–Marquardt training algorithm.

4.3.1.3 Activation function

Another important point in the configuration of the network before training it is the activation functions of the neurons in both the hidden and output layers. In this case, for the neurons in the hidden layer the hyperbolic tangent function is used because it provides better results when the Bayesian regularization is used (Dan Foresse & Hagan, 1997). On the other hand, for the neuron in the output layer the linear activation function is used because, as said in Section 4.2.2, a network with a hidden layer with sigmoid (or hyperbolic tangent) neurons and an output neurons with linear activation function can approximate any continuous function. The hyperbolic tangent activation function is limited within the range [-1, 1]. Thus, usually, the training data is normalized in the same range. In neural networks training, the normalization of the data is an important task. For example, without normalized data, some inputs can have much higher values than others and, consequently, the higher the value of the input, the higher the effect over the training of the network. However, it is not strictly necessary for a good performance of the net (Wu & and Lo, 2010).

4.3.1.4 Weight and biases initialization

Besides, in order to yield better performance of the net is usually recommended to train the network several times changing the initial weights and biases so as to avoid local minimum. In this work, instead of applying random initialization of the initial weights and biases, the Nguyen-Widrow initialization method is used (Nguyen & Widrow, 1990). The main idea behind this initialization is choosing the weights so as the hidden units are scattered in the input space X. Thus, doing this the learning speed of the network with multiple inputs improves (Nguyen & Widrow, 1990).

4.3.2 Training and testing dataset configuration

This subsection explains the procedure followed to prepare the training and testing database. First, the different implications of the pre-processing of the experimental database are described and argued. Second, the selection and generation of specific training and testing examples is also explained.

4.3.2.1 Training dataset pre-processing

Another important step before training the network is the pre-processing of the data in order to extract the most representational accuracy from the experiments.

Thus, in order to obtain periodic dynamic evolutions, in each experiment interpolation techniques have been applied to the sequence of measurements of the specific grinding energy. In particular, the *smoothing spline* with a *smoothing parameter* value equal to 1.91 is applied so as to avoid excessive oscillations. However, not all the dynamic evolutions have the same number of points (time-steps) because some wheels are softer and, thus, they wear faster. Therefore, those examples with less points (time-steps) are filled with a *NaN* value provided by Matlab™. Thus, all the examples have the same number of points but during the training of the net the software does not take into account these *NaN* values.

Alphabetical letter	Number	Alphabetical letter	Number
A	0	N	0,52
B	0,04	O	0,56
C	0,08	P	0,6
D	0,12	Q	0,64
E	0,16	R	0,68
F	0,2	S	0,72
G	0,24	T	0,76
H	0,28	U	0,8
I	0,32	V	0,84
J	0,36	W	0,88
K	0,4	X	0,92
L	0,44	Y	0,96
M	0,48	Z	1

Table 5 Transformation of grit size letters into numbers

Finally, the grit size is denoted using alphabetic letters (Section 2.1). However, in this case in order to have a numerical representation required to train RNN, these alphabetic letters are transformed into numbers. Thus, the letters from ‘A’ to ‘Z’ are converted into numbers from 0 to 1 (Table 5).

4.3.2.2 Selection of the training and testing examples

In this work each experiment constitutes one training sample i.e. each complete dynamic evolution is one training sample. Thus, there are 46 samples (Figure 21). Since the Bayesian regularization is used, in this case it is not necessary to use validation data. Therefore, the data can be divided into training and testing dataset. This is very useful when there is a lack of experiments for training the net. The training and testing data are shown in Figure 24.

As said before, one of the objectives of the presented methodology is to generalize to new grinding wheels. Thus, one of the wheels cannot be used for training the net. In particular, the two experiments carried out with the 82AA36G6VW grinding wheel are used to test the generalization capability of the net to new grinding wheels not used during the training phase of the RNN. On the other hand, it is also analysed the generalization capability for new cutting conditions for a specific wheel not used during the training (note that these cutting conditions are used for other wheels in the training database). In total, 42 examples are used for training and 4 for testing (Table 6).

Figure 24 Training (orange) and testing (green) data

	Grit size	*Hardness*	q_s	Q'*(mm³/mm·s)*
Test experiment 1	36	K	100	2.5
Test experiment 2	70	G	60	1
Test experiment 3	36	G	60	2.5
Test experiment 4	36	G	60	4

Table 6 Test experiments used for testing the generalization capabilities of the net

Due to the lack of samples to train the network, virtual experiments are generated. In this case, 10 experiments are generated from each of the 42 training experiments. Thus, the data for training are 420 dynamic evolutions. However, the data for testing are still four, because there is no point in generating virtual data for testing. For generating these virtual experiments, the approach is based on the variability of the measuring process in the real world of the wheel wear, surface roughness and specific grinding energy for generating the training database. This is mainly caused by its stochastic nature due to aspects such as the hand-made production of the wheels or the system stiffness.

Thus, the virtual experiments are generated within the ±10% range for the specific grinding energy. However, for the wheel wear and the surface roughness the virtual experiments are generated within the ±5% because the variability in these cases is lower than for the specific grinding energy. The decision of these percentages is based on the experience of the potential users of the sensors. Thus, in Figure 25 the 10 virtual experiments of specific grinding energy generated from a real one are shown.

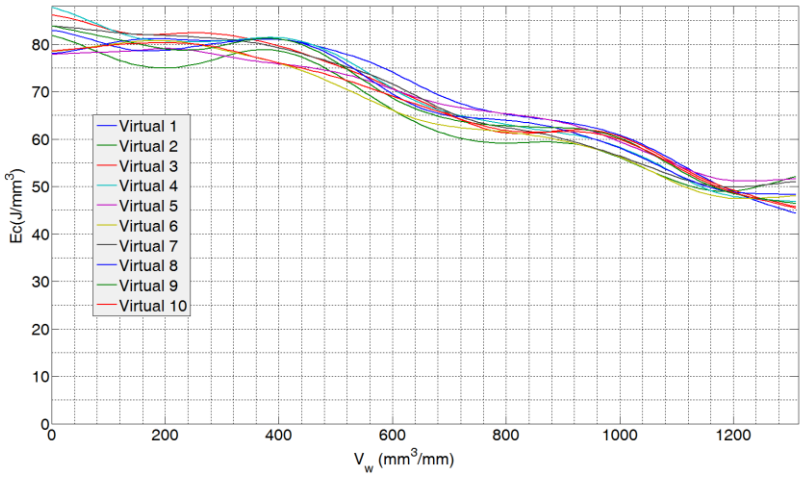


Figure 25 Virtual specific grinding energy signals versus specific volume of material removed

Finally, the inputs of the net such as wheel characteristics and grinding conditions are constant values. However, in each time-step all the inputs of the network need to have data. Thus, it is necessary to convert these constant values into constant evolutions with as many points as the dynamic evolution signal has (Figure 26).

Figure 26 The constant inputs are converted into constant evolutions (hardness, q_s, Q' and grit size) to cover all the prediction horizon of the specific grinding energy (E_c)

First, it is necessary to convert the static training inputs (the grit size **G**, the hardness **H**, the speed ratio **S** and material removal rate **Q**) into constant evolutions. Therefore, vectors with static values are generated for each training input (representing different wheel characteristics and cutting conditions):

$$\boldsymbol{G} = [G(1)\ G(2) \dots G(k-1)\ G(k)]$$

$$\boldsymbol{H} = [H(1)\ H(2) \dots H(k-1)\ H(k)]$$

$$\boldsymbol{S} = [S(1)\ S(2) \dots S(k-1)\ S(k)]$$

$$\boldsymbol{Q} = [Q(1)\ Q(2) \dots Q(k-1)\ Q(k)]$$

where *k* is the number of points in the time series.

From **G**, **H**, **S** and **Q** vectors, a submatrix is generated per training sample (1):

$$i_j(p) = [G(p)\ H(p)\ S(p)\ Q(p)]^T \qquad \text{(x)}$$

where *j* is the specific train sample (j ∈ [1-n]), n is the number of training samples and *p* is the specific point in the constant evolution (p ∈ [1-k]). Thus, the training input matrix (*I*) is as follows:

$$\begin{vmatrix} i_1(1) & i_1(2) & \dots & i_1(k\text{-}1) & i_1(k) \\ i_2(1) & i_2(2) & \dots & i_2(k\text{-}1) & i_2(k) \\ \dots & \dots & \dots & \dots & \dots \\ i_{n-1}(1) & i_{n-1}(2) & \dots & i_{n-1}(k\text{-}1) & i_{n-1}(k) \\ i_n(1) & i_n(2) & \dots & i_n(k\text{-}1) & i_n(k) \end{vmatrix}$$

Each training output sample is a unique dynamic evolution of *k* points representing the target signal generated with different grinding wheels under different cutting conditions. Thus, the training output array (**O**) is as follows:

$$\begin{vmatrix} o_1(1) & o_1(2) & \dots & o_1(k\text{-}1) & o_1(k) \\ o_2(1) & o_2(2) & \dots & o_2(k\text{-}1) & o_2(k) \\ \dots & \dots & \dots & \dots & \dots \\ o_{n-1}(1) & o_{n-1}(2) & \dots & o_{n-1}(k\text{-}1) & o_{n-1}(k) \\ o_n(1) & o_n(2) & \dots & o_n(k\text{-}1) & o_n(k) \end{vmatrix}$$

where *n* is the number of training samples and *k* is the number of points in dynamic evolution.

As mentioned above, supervised learning is used to train the RNN. In fact, for each training input sample (i_j) the desired output (o_j) values are presented. Thus, as batch training is used, only after all the inputs (**I**) and targets (**O**) samples are presented, the weights and biases of the net are updated.

4.3.3 The best network selection

After configuring the network and the training process, the next step is the selection of the best network i.e. the network that best models the non-linear relationship between the inputs and the output of the ANN. To do this, the best network structure must be selected. These structures differ from the number of neurons (HN) in the hidden layer and number of delays (D) in the feedback.

If the configuration of the network and the training process is based on the knowledge of the process extracted from the analytical models, the selection of the best network structure is based on a comparative study of different HN-D combinations.

4.3.3.1 Two-phases methodology

Thus, the best structure selection is carried out varying the number of neurons in the hidden layer (HN) and delays in the feedback and comparing the results with the trial-and-error approach to select the net with the lowest error. The selection of the network is divided into two phases:

1. Coarse tuning: during this phase the aim is to obtain the network structure (HN-D) around which the lowest test errors are yielded.
2. Fine tuning: the fine tuning of the training process is to be performed in the range given by the best structures inferred in the coarse tuning. Considering the fine nature of this phase, the values of HN and D of the new configurations to be trained are taken one by one within the specified range.

Each training of the RNN with a particular network structure is carried out *m* times to avoid a local minimum. As said in Section 4.3.1.4 for the initialization of the weights, the Nguyen-Widrow algorithm is applied.

4.3.3.2 Indicators

As said in Section 4.3.2.2, the accuracy of a trained network structure is computed on another subset of grinding experiments, the so-called test experiments, through the following indicators that provide measurements of the deviation between the real value of the wheel wear and the value provided by the ANN-based model (see Figure 27):

1. The mean square error MSE: it is provided by the Matlab^{TM} training tool and, therefore, its value is given within the normalization range.

2. The Mean Absolute Maximum Error (MAME): It is calculated in units of process (micrometers—μm) as:

$$MAME = \frac{1}{n}\sum_{x=1}^{n}|\max(T_x - P_x)| \quad (13)$$

where T is the target of the model, P is the predicted value and n is the number of test examples.

Since these indicators are computed for the testing dataset, they allow to quantify the generalization capability of the RNN.

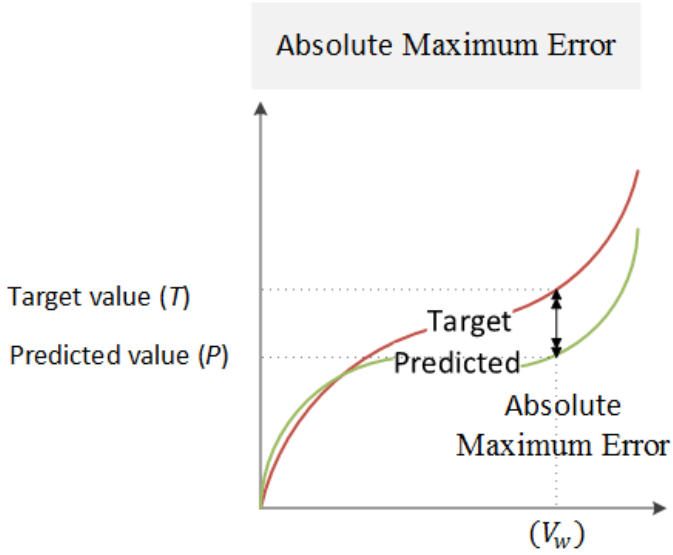


Figure 27 The absolute maximum error for one test. After, the mean of all test experiments is computed with the MAME indicator

This MAME metric is used instead of the widely used for evolution analysis Mean Absolute Percentage Error (MAPE) because from the grinding process point of view it is easier to measure the error using absolute errors instead of percentage ones. In fact, in advanced manufacturing processes, the surface roughness error, for example, is usually given in absolute error. Therefore, in this work this point of view has been followed.

4.4 Conclusions

This chapter deals with the general methodology for modelling dynamic evolutions of grinding signals. In this case, this approach is focused on modelling the dynamic evolution of the wheel wear, the surface roughness and the specific grinding energy. This

methodology is based on the knowledge about the grinding process extracted from analytical models widely recognized.

These analytical models relate the specific grinding energy, the wheel wear and the surface roughness between them and with the wheel characteristics and cutting conditions. Thus, the main conclusion extracted from this relationship is that the specific grinding energy and the surface roughness are dependent on the state of the wheel. Therefore, the behavior of the specific grinding energy and the surface roughness signals change while more parts are machined.

As a result of the analysis of the analytical models, the inputs of the ANN model are selected. These have to represent the wheel characteristics and the grinding conditions. Therefore, based on the application field selected, grit size and wheel hardness are chosen to characterize the grinding wheel while specific material removal rate (Q') and speed ratio (q_s) are used to characterize the grinding process.

Concerning the selection of the ANN architecture, the feedforward ANN architecture is discarded because it is not possible to model the dynamic evolution with static inputs with feedforward neural networks, as it is the case of the most restrictive case of the application field of this work, i.e. the estimation of the specific grinding energy. Therefore, among the classical architectures, the only way of modelling the dynamic evolution with static inputs is using Recurrent Neural Networks. Among different RNNs the Elman based LRNN is selected because it functions without any initial output values, highly important based on the conclusions of the Chapters 2 and 3.

Besides, other decisions such as training algorithm, generalization strategy, activation function or weights and biases initialization are decided based on the state of the art. As a training algorithm the well-known Levenberg-Marquardt training algorithm is selected for its fast convergence. Then, it is highly important to select how to improve the generalization capabilities of the net. For the present work, the Bayesian regularization is selected for its good generalization capabilities and because there is no need to divide the database into three different datasets (training, validation and testing). Thus, more data can be used for training if there is lack of samples. Finally, the hyperbolic tangent activation is chosen as activation function of the hidden neurons and Nguyen-Widrow approach for initialization of the network weights and biases to improve the learning speed.

Once the network architecture is selected and configured, the next key point is the training and testing database configuration. First, the data is pre-processed in order to obtain periodic dynamic evolutions with *smoothing spline* interpolation, adding *NaN* value provided by Matlab™ at the end of the examples with less points. Thus, all the examples have the same number of points. Then, the training and testing datasets are selected. From the 46 experiments available, 4 are selected for testing the generalization capabilities of the trained network. Among all the experiments, two to test the generalization to new grinding conditions and other two to test the generalization to new wheel characteristics are selected. Finally, due to the lack of samples to train the network, virtual experiments are generated. In this case, 10 experiments are generated from each of the 42 training experiments.

The last part of the presented methodology is how to select the best network, the network that more precisely predicts the dynamic evolution for the test experiments. Thus, in this work a two-phase methodology is proposed. First, the phase called “coarse tuning” is done. During this phase the aim is to obtain the network structure (HN-D) around which the lowest MSE test errors are yielded. Given the best structures inferred in the coarse tuning, the so called “fine tuning” is carried out in HN-D structure taken one by one within the specified range. During the fine tuning, the MAME metric is used instead of the widely used for evolution analysis Mean Absolute Percentage Error (MAPE) because from the grinding process point of view is easier to measure the error using absolute errors instead of percentage ones.

5 SMART SENSORS FOR GRINDING PROCESS USING RNN

In grinding monitoring grinding wheel wear and the workpiece surface roughness is essential nowadays in order to improve the performance of the machines. Being really difficult to measure on-line the wheel wear and the surface roughness, another strategy to measure these grinding variables is the no-direct method. Therefore, the aim is to develop two different soft sensors for measuring wheel wear and surface roughness, respectively, without introducing any sensor in the machining area. To do so, the ANN based strategy described in Chapter 4 is used. The measurable signal used is the specific grinding energy. This signal is an easily measurable variable without introducing any sensor in the machining area. New indicator is introduced because it is not possible to select the best network only with MSE and MAME. Thus, the coefficient of variance is proposed to analyse not desirable phenomenon observed in the last part of the predictions. The proposed methodology has shown the potential for modelling the dynamic evolution of the wheel wear and surface roughness without measuring initial real values in a prediction horizon up to 2000 mm^2 of specific volume of part material removed. In fact, the MAME error for wheel wear is 32 µm, and 0.26 µm for surface roughness.

Monitoring industrial manufacturing processes is essential nowadays in order to improve the performance of the machines. Thus, in grinding two of the most important variables to be monitored are the grinding wheel wear and the workpiece surface roughness. As said in Section 2.1, knowing these variables is extremely important because they provide information about the cutting ability of the wheel and the surface quality of the part. Actually, monitoring on-line (during the operation) the wheel wear can lead to extend the lifecycle of the wheel before dressing it to recover its cutting ability with the advantage of not only increasing the production, but also avoiding workpiece burnings, one of the most common thermal damage (Malkin & Guo, 2008). Regarding the surface roughness of the machined parts, it can only be measured once the machining process is finished. Thus, the measurement of the surface finish cannot be accomplished on-line, i.e. during

the machining process. Besides, as said in Section 1, monitoring the wheel wear and the surface roughness is harshly limited by the low accessibility of the machining area. In fact, the high rotational speed of the grinding wheel (common values are around 45 m/s, but there are industrial examples of as much as 200 m/s), the generation of abrasive swarf, the presence of large quantities of coolant at high pressures, etc. all combine to limit the possibility of using industrial sensors during the process.

Being really difficult to measure on-line the wheel wear and the surface roughness, another strategy to measure these grinding variables is the no-direct method. Actually, using soft sensors with inputs that are easily measurable output-related grinding variables together with properly applied intelligent techniques allow to infer the desired measurable variables. Therefore, the aim is to develop two different soft sensors for measuring wheel wear and surface roughness, respectively, without introducing any sensor in the machining area. To do so, the ANN based strategy described in Chapter 4 is used.

Hence, the Chapter's layout is as follows. First, the inputs and the outputs of the smart sensors are described. Second, the specifically required pre-processing of the training data is explained. Third, the comparative study and results of both sensors (wheel wear and surface roughness) are shown. Finally, the conclusions are addressed.

5.1 Inputs and outputs of the smart sensors

As said in Section 1, the aim of developing soft sensors is to measure on-line a variable not easy to measure directly. In fact, another easy measurable variable is used and then the target variable is inferred using the ANN strategy. Therefore, guided by the analytical models used for developing the ANN based strategy (Chapter 3), it can be noticed that the surface roughness and the wheel wear are closely linked with the specific grinding energy. Besides, as said in Section 4.2.3.2, the specific grinding energy is an easily measurable variable.

Hence, for developing the two soft sensors, besides the ANN inputs described in Section 4.2.1, the specific grinding energy is used as input. Consequently, the ANNs for soft sensor have 5 inputs.

5.1.1 Training and testing dataset configuration

Usually, before using any sensor some parameters of the sensor must be adjusted. This phase is usually named as the calibration of the sensor. In the same way, the calibration

of the ANN-based smart sensor is analogous to the training of the network in order to find the best possible network parameters such as the number of neurons in the hidden layer and the number of delays in the feedback.

Thus, this subsection explains the procedure followed to prepare the training and testing database for the calibration of the soft sensor. First, the different implications of the pre-processing of the experimental database are described and argued. Second, the selection and generation of specific training and testing examples is also explained.

5.1.1.1 Pre-processing

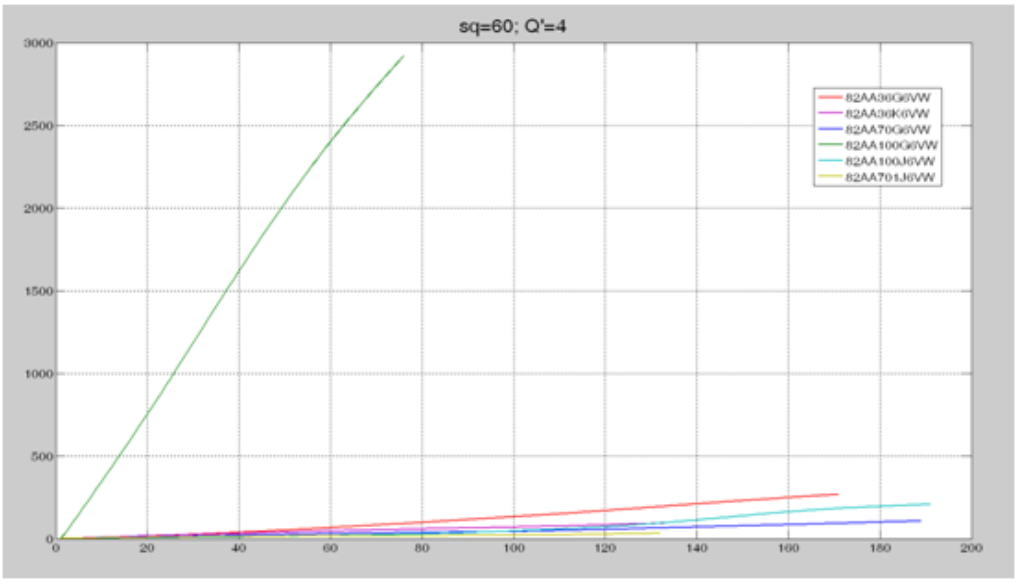
As stated before, an important step before training the network is the pre-processing of the data in order to extract the most representational accuracy from the experiments.

As explained in the ANN based strategy in Section 4.3.2.1, in order to obtain periodic dynamic evolutions, in each experiment the *smoothing spline* interpolation has been applied to the sequence of measurements of surface finish and wheel wear. However, during the design of the soft sensor, one important characteristic is the sampling frequency i.e. how often a measured value is provided. In this case, the criterion is to yield one value every specific material removal (V'_w) equal to 10 mm^3/mm, because in grinding the phenomena are progressive and, thus, there is no need to use high frequency sensors. On the other hand, from the process point of view it is not recommendable to develop a sensor with very slow frequency because relevant information is lost. Besides, the same interpolation method and criteria is used for the specific grinding energy because the number of points of the input data and output desired data must be the same.

Finally, another important aspect to analyse is the existence of considerable differences between the values of a specific input/output in the experimental training set. In this work, this is the case of the wheel wear experimental tests, as shown in Figure 28(a). It can be noticed that one of the experiments yields very high values compared to the rest of the experimental tests. The use of these data just normalized to the range of the activation function can affect negatively to the results of the training process due to the great difference of magnitude between one experiment and the others (Chandana Prasad & Beg, 2009). Actually, it can make difficult to learn the cases with similar magnitudes due to the decrease in the resolution that suffer most of the experimental data as a consequence of including the experiment with a much higher wheel wear. In order to avoid this effect, a natural logarithmic processing of the wheel wear data is applied to decrease such great differences, as shown in Figure 28(b).

Logarithmic processing

(a)



(b)

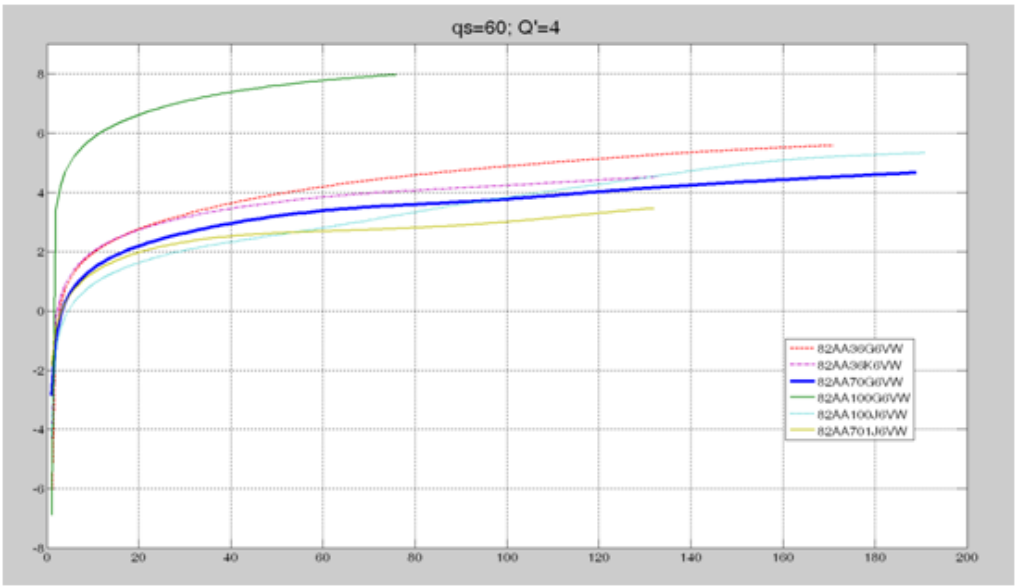


Figure 28 (a) Original wheel wear data. (b) Wheel wear data after logarithmic processing

This way, the data after the logarithmic transformation is closer and the difference between high and low data values is narrower. Of course, one important point to take into account is that the logarithmic value of zero is not defined. In order to deal with this matter, the initial zero value has been replaced by a very low value close to zero. However, it only affects the initial point as Figure 28(b) shows.

5.2 Comparative study and results

As said before, two different soft sensors are developed in this work. One for measuring the wheel wear and another one for measuring the surface roughness. Therefore, the calibration (selection of the best possible number of neurons in the hidden layer and number of delays) of both sensors has to be done separately. Thus, first, the soft sensor that measures the wheel wear during the process is calibrated and, second, the calibration is applied on the sensor for the surface roughness.

5.2.1 Wheel wear

As aforementioned, the aim is to develop a soft sensor capable to provide during the process the wheel wear, ergo, the state of the wheel. Thus, in this Section, the results achieved and a discussion about those results are presented. First, a comparison analysis related to the network structure is presented. These structures differ from the number of neurons (HN) in the hidden layer and the number of delays (D) in the feedback. Second, the results related to the soft sensor and its capability to model the dynamic evolution of the wheel wear are shown. Finally, the ANN that better represents the wheel wear is presented.

5.2.1.1 Analysis of Results

As stated in Chapter 3, firstly, coarse and fine tunings are performed with both training and testing datasets (Section 4.3.3.1). The coarse tuning aims at defining the dimension and the dynamic behaviour of the ANN. The aim is to obtain the structure (HN-D) around which the lowest test errors are yielded.

Regarding the results, the lowest test MSE values yielded by the ANNs trained with different combinations of hidden neurons in the hidden layer (HN) and delay units in the feedback (D) are presented in Figure 29. When the delay value is 5 the lowest MSE values are within the range 5–8 HN but, nevertheless, for a delay equal to 10, the range shifts to 7–10 neurons in the hidden layer. Finally, although the HN5D15 network configuration yields a very low MSE value, from 7 HN the MSE error increases significantly (actually, it can be noticed that the error increases linearly, i.e. faster than in the other cases). Thus, it is concluded that within the ranges [5, 10] neurons in the hidden layer and [5, 10] delay units are to be found the proper configuration to model the wheel wear behaviour. Actually, increasing the number of hidden neurons and/or delay units provides higher test

errors that can be associated to the overfitting of those ANNs. Moreover, the additional disadvantage is requiring much longer training times.

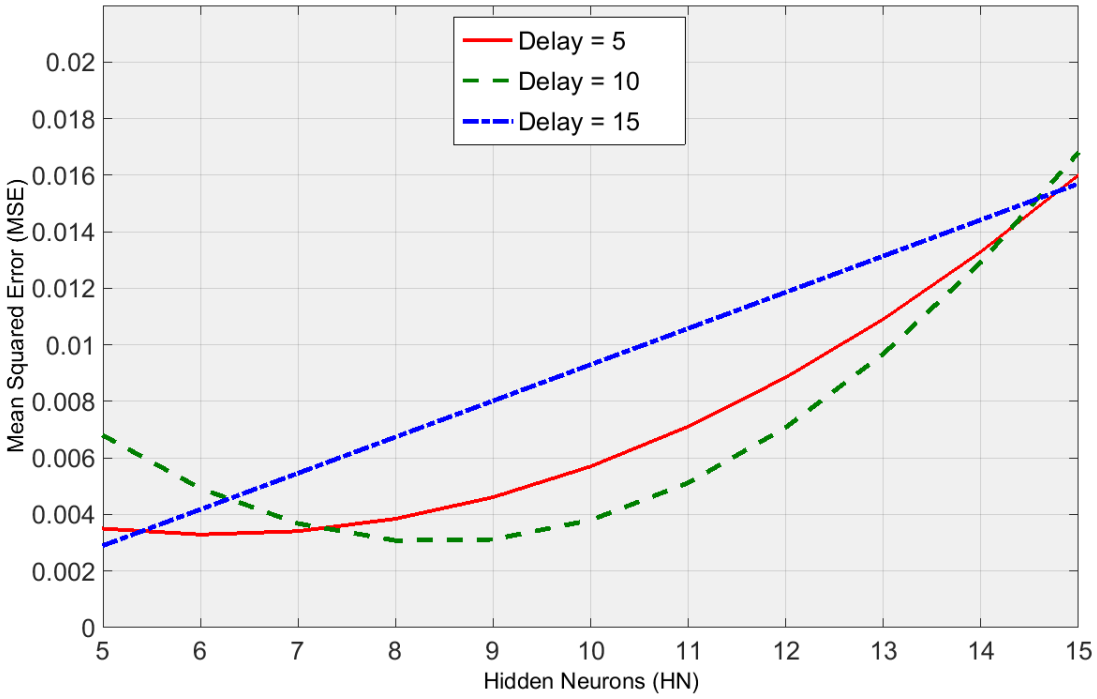


Figure 29 Summary of the results of coarse tuning

Then, the *fine tuning* of the training process is performed between 5 and 10 hidden neurons in the hidden layer and 5 and 10 delay units. As a summary of the *fine tuning* results, the ANNs that provide the average of the three lowest error predictions for the testing dataset obtained in the fine tuning phase are presented in Table 7. Given MSE values of the same order of magnitude, more than one ANN are preselected so as to analyse the behaviour of the best neural structures from the grinding process perspective. Likewise, it can be noticed that all the test MSE values are lower than the best one obtained in the preliminary analysis. The more trainings performed around the proper dimension and timing parameters *HN* and *D*, the more probability to reach lower MSE values. However, the best ANN from the grinding process perspective is not the one with the lowest MSE, i.e. the network net3 with structure with 10 neurons in the hidden layer and 10 delay units (10HN10D) provides the lowest values of MAME.

Hidden Neurons (HN)	Delays (D)	net	MSE	MAME (µm)
9	8	Net3	0.00177	43
10	10	Net3	2.34e-04	32
5	5	Net2	2.17e-04	53

Table 7 Summary of the results of fine tuning

In order to illustrate the results of the Table 7, Figure 30 to Figure 33 show the behaviour of the preselected ANNs for the four test experiments of the testing dataset, as well as the target or real wheel wear values. Figure 30 shows the evolution of the estimation of wheel wear for the three preselected ANNs in the test experiment 1 82AA36K6VW q_s = 100; Q' = 2.5 (the prediction for grinding conditions for known grinding characteristics). In this case, similar good performance is achieved by the three ANNs; actually, MAME is 13, 9 and 8 µm for 9HN8D net3, 10HN10D net3 and 5HN5D net2, respectively, which are considered very good estimations from the grinding process perspective. On the contrary, in the rest of the test experiments the differences of performance of the preselected ANNs are more evident.

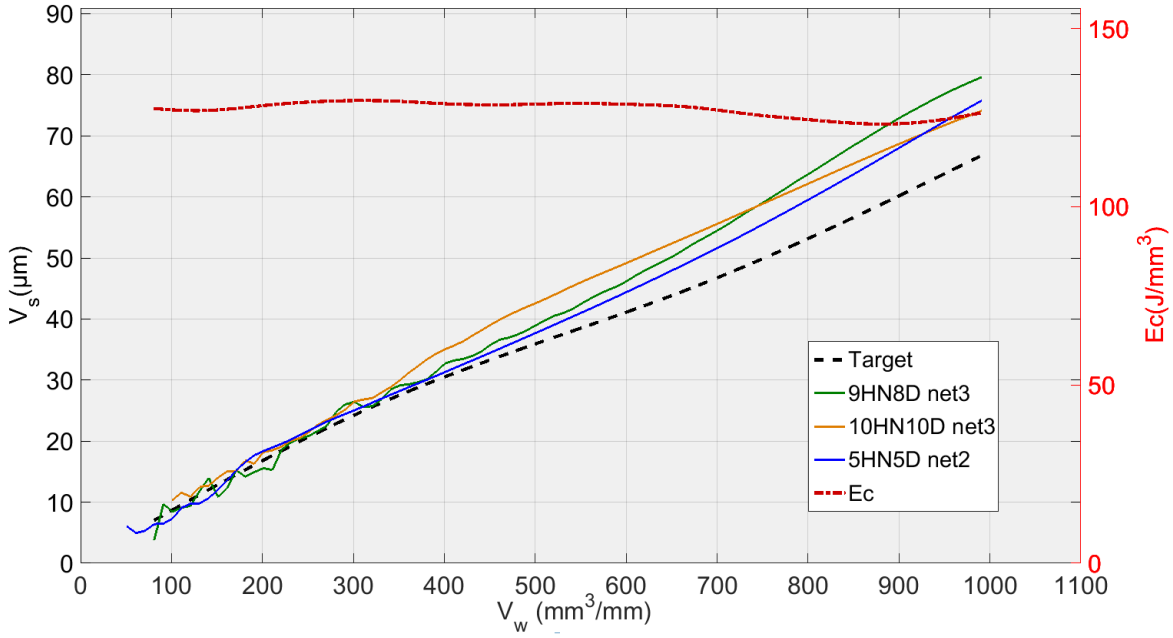


Figure 30 Wheel wear generalization capability of the three preselected structures. Test experiment 1: 82AA36K6VW qs = 100; Q' = 2.5

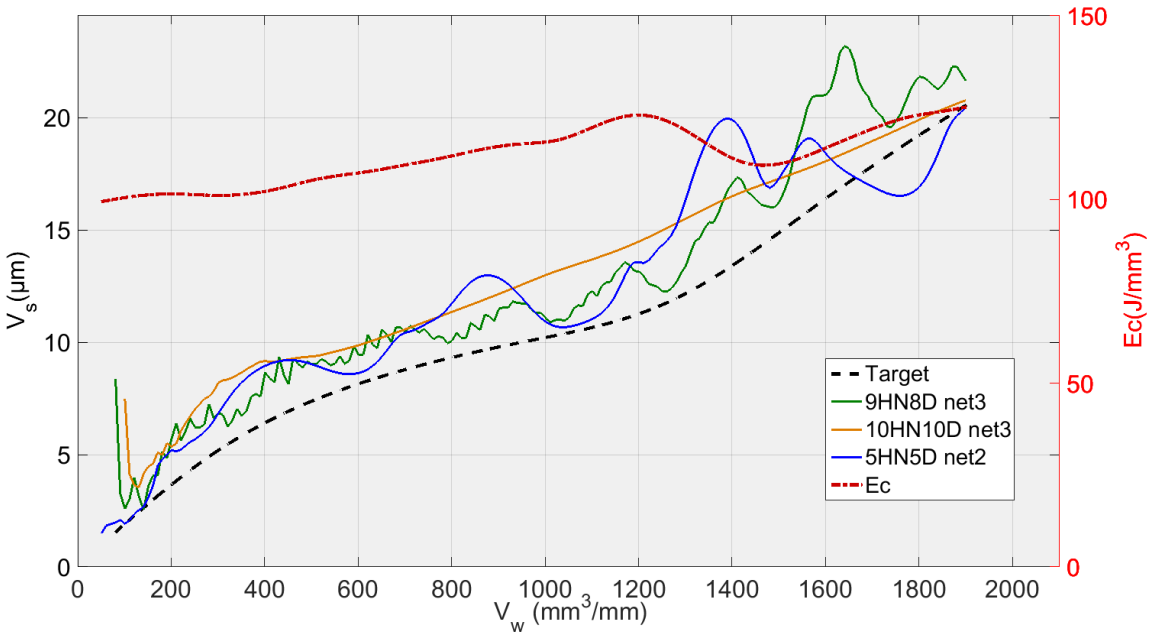


Figure 31 Wheel wear generalization capability of the three preselected structures. Test experiment 2: 82AA70G6VW qs = 60; Q' = 1

In the case of the second test experiment (the prediction for grinding conditions for known grinding characteristics), Figure 31 shows that the best behaviour is provided by 10HN10D net3. Actually, 9HN8D net3 and 5HN5D net2 provide an oscillatory behaviour of the output with an AME value of 6 µm in both cases, while in the case 10HN10D net3 is 3 µm. Although from the grinding process perspective an error of 6 µm is considered very low in terms of estimation, the oscillatory behaviour of the estimations can indicate a lower generalization capability of the ANNs. Actually, the behaviour of the estimations of the wheel wear remains constant at the end in 9HN8D and 5HN5D for the third (Figure 32) and fourth (Figure 33) test experiments (82AA36G6VW q_s = 60, Q' = 2.5, and 82AA36G6VW q_s = 60, Q' = 4), while in the case of 10HN10D there is not any kind of constant behaviour. Moreover, 10HN10D network structure provides the lowest MAME value for the third and fourth test experiments, which are 49 µm and 67 µm, respectively.

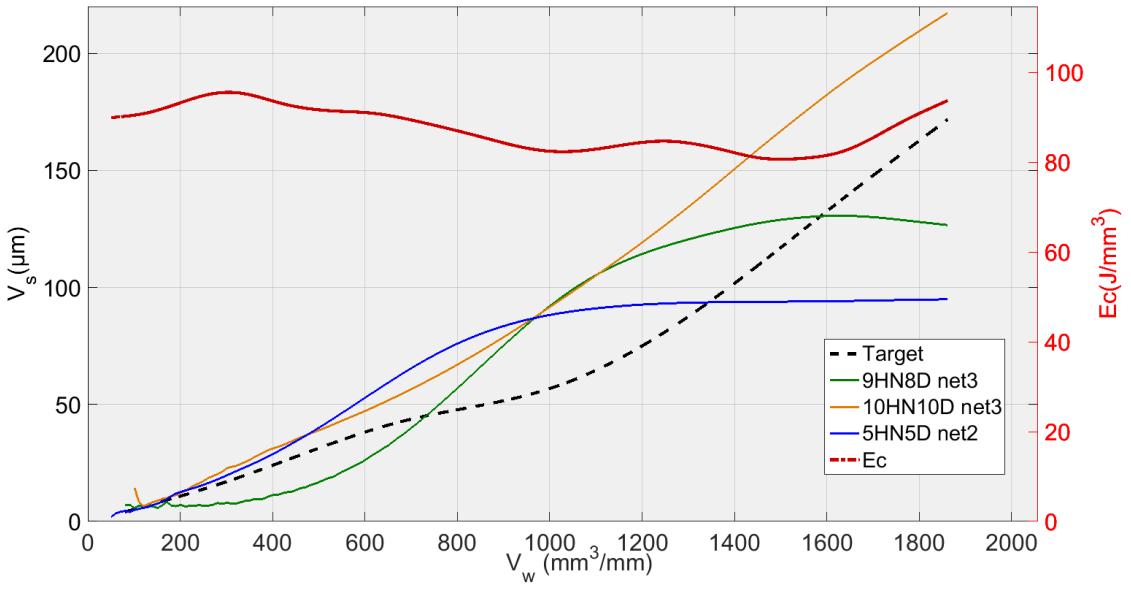


Figure 32 Wheel wear generalization capability of the three preselected structures. Test experiment 3: 82AA36G6VW qs = 60; Q' = 2.5

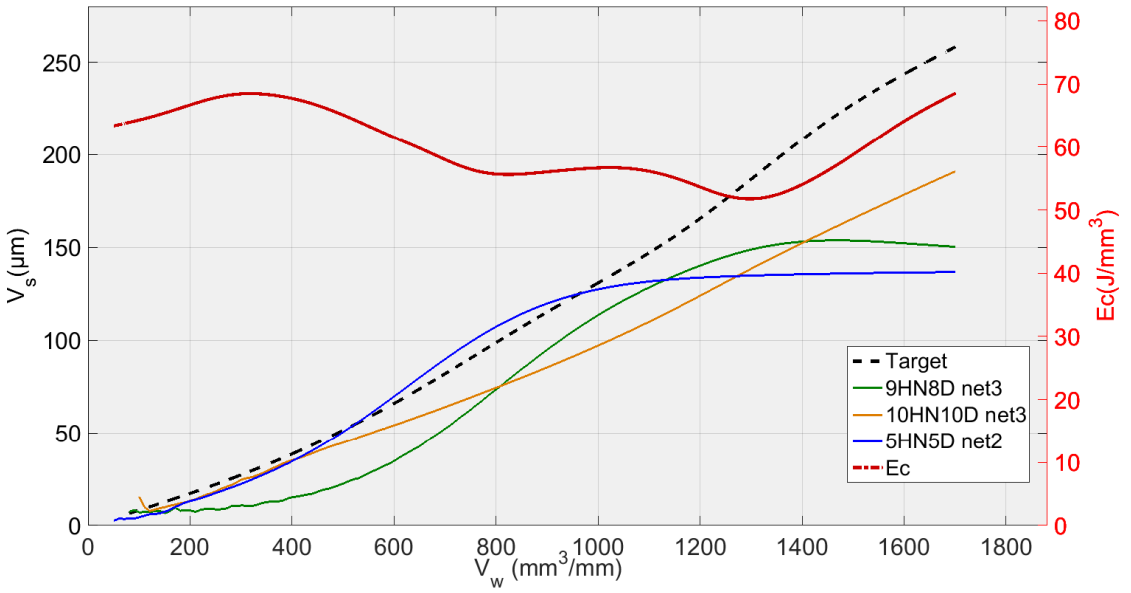


Figure 33 Wheel wear generalization capability of the three preselected structures. Test experiment 4: 82AA36G6VW qs = 60; Q' = 4

Thus, it can be concluded that the RNN 10HN10D net3 provides the most balanced performance, which agrees with the corresponding values of MAME provided in the results of Table 1. Thus, among the three preselected neural structures shown in Table 1, the network 10HN10D net3 has been selected, *i.e.*, an ANN based on the Layer-Recurrent Neural Network architecture with one hidden layer consisted of 10 hidden units and with a memory equivalent to 10 delays.

From the previous analysis it has been observed that in some cases the predicted signals remain constant in the final part of the prediction. This is a non-desirable phenomenon because the aim is to predict the real wheel wear signal tendency as close as possible for the complete dynamic evolution of the wheel wear. Therefore, a new metric is used to analyse the precision in the final stages of the dynamic evolution, where this phenomenon is observed. In fact, the coefficient of variation (CV) (Equation 13) in the last 4/5 part of the signal is calculated to measure this phenomenon.

$$CV = \frac{\sigma}{\mu} \tag{14}$$

where σ is the standard deviation and μ is the mean. This metric allows to observe the phenomenon in a relative manner allowing for comparisons.

This metric is used in the final stage of the analysis of the results, when the best networks are selected using the MSE and the MAME.

	Test	*9HN8D net3*	*10HN10D net3*	*5HN5D net2*
Coefficient of Variation (CV)	Test experiment 1	0.0691	0.0512	0.0749
	Test experiment 2	0.0529	0.0527	0.0640
	Test experiment 3	0.0093	0.0779	0.0035
	Test experiment 4	0.0071	0.0755	0.0032

Table 8 Results of the coefficient of variation analysis for the 4/5 of the prediction horizon of wheel wear

In Table 8 the coefficient of variation results for the last 4/5 part of the signal are shown. The results show that high and similar CV values are yielded for all the nets for the test experiments 1 and 2. However, for the test experiments 3 and 4 there are differences for on one hand 10HN10D net3 and on the other hand 9HN8D net3 and 5HN5D net2 network. Actually, the CV results yielded for the 10HN10D net3 are higher, similar to those yielded for the test experiments 1 and 2. This is consistent with the results shown in Figure 30-33. In particular, it is noteworthy to examine the case of the test experiment 3. In this case, the CV value is low, except for the 10HN10D net3 network. Actually, in Figure 32 it can be observed that the signal of the 10HN10D net3 is the only one that does not remain constant at the end. Besides, for the test experiment 4, it has, also, the highest CV value. Regarding the 5HN5D net2 network, it shows the lowest coefficient of variation value for the test experiments 3 and 4. Actually, the Figure 32 and Figure 33 show that the signals at the end of the prediction horizon are practically constant. Finally, the network 9HN8D net3 shows, also, close to constant behaviour for the test experiments

3 and 4 with a low CV value, very low value compared to the values yielded by the 10HN10D net3 network.

Therefore, the CV analysis also confirms that the network with a better behaviour is the 10HN10D net3 network. Besides, the coefficient of variation analysis can be useful to detect the constant behaviour of the signal at the end of the prediction horizon.

On the other hand, from a quantitative point of view, the results show that the initially proposed time horizon (equivalent to 2000 mm^3/min) is very ambitious. Thus, it is necessary to delimit a time horizon in which the ANN-based sensors behave reliably. Concerning the ANN-based sensor for the estimation of wheel wear, MAME ranges in the whole prediction horizon from 3 to 67 μm, while for a prediction horizon equivalent to 600 mm^3/min this range reduces drastically to 3–12 μm, which is very accurate from the grinding process perspective. Also notice that during the prediction horizon equivalent to 600 mm^3/min, 390 parts are machined, which is a significant quantity. Nevertheless, from a qualitative point of view the output of the ANN-based sensor tracks the trend of the actual values during the whole prediction horizon. Concerning the *constancy* phenomenon, notice that the selected network for the estimation of wheel wear (structure 10HN10D: 10 hidden neurons in the hidden layer, 10 delay units) does not exhibit that phenomenon in any of the test experiments (*i.e.*, experiments with characteristics not considered during the training process).

Additionally, the results show that better performance is achieved when the virtual sensor estimates the behaviour of a wheel employed during the training process, but under new cutting conditions (see Figure 30 and Figure 31). Actually, the performance of the virtual sensor decreases when predicting the behaviour of a new wheel (a wheel not used during the training process), but under known cutting conditions (see Figure 32 and Figure 33). This means that the characteristics of the wheel have a higher influence on the wear behaviour than the cutting parameters.

A final remark can be done from the point of view of industrial application. For many users and also for many grinding wheel manufacturers, a single numerical indicator is used to analyse the performance of the wheel from the point of view of wear over time. This is known as the grinding ratio (G), which is defined in Equation 14:

$$G = \frac{\Delta V_w}{\Delta V_s} \tag{15}$$

This parameter can be additionally used for assessing the quality of the tuning process of the new sensor. When looking at wear patterns along time, an initial short stage during which the wear rate is very fast can be observed. Afterwards, it is accepted (Marinescu, et al., 2004) that a linear behaviour prevails, and it is during this stage that *G* is measured. Therefore, grinding ratio values obtained from actual wear measurement during experimental tests and those produced by the virtual sensor have been calculated and compared. Results have been collected in Table 9.

Test experiments	*Experimental*	*Sensed*
Test experiment 1	16	14
Test experiment 2	100	110
Test experiment 3	10	8
Test experiment 4	6	8

Table 9 Summary of grinding ratio (G) results of the test experiments

Table 9 shows the grinding ratio (*G*) results of the four *test experiments*. The results are consistent with the wheel wear results and show that the best ones are achieved with the wheels used during training but under new cutting conditions. For the first and second *test experiments* the errors are 2 and 10, respectively. These errors are very satisfactory from an expert on grinding point of view showing the potential of the presented sensor. In the case of a new wheel (third and fourth *test experiments*), the error is 2 for both *test experiments*. The absolute errors are considered low for the grinding process perspective in all the cases, but the relative ones are a bit higher for the third and fourth *test experiments.* Therefore, as mentioned above, it can be clearly seen that the characteristics of the wheels have higher influence on the wheel wear prediction and, consequently, on the generalization capabilities of the ANN-based sensor.

Finally, at the sight of the results it is clear that the best network is the one with 10 neurons in the hidden layer and 10 delay units. It has 5 inputs, 10 neurons in the hidden layer, 10 delay units in the feedback from the output of the hidden layer to the inputs of the hidden layer, and, finally, one output.

5.2.2 Surface Roughness

In this case, the aim is to develop a soft sensor capable to provide measurements of the surface roughness under the influence of the wheel wear. Thus, similar to the previous Section, the results achieved and a discussion about those results are presented. First, a discussion about the implications regarding the structure is provided. Second, the comparison analysis related to the network structure is presented. Third, the results related

to the soft sensor and its capability to model the dynamic evolution of the surface roughness are shown. Finally, the ANN that better represents the surface roughness is presented.

It should be recalled that, in order to develop the ANN-based sensor for measuring the surface roughness, the followed steps are the same that for developing the sensor for wheel wear. Thus, the network structure is selected with the same ad-hoc metrics used for the wheel wear modelling.

Another important point related with the surface roughness is that there is no need to use the natural logarithmic pre-processing for the surface roughness data because the differences in the experimental values are not significant in this case.

5.2.2.1 Structure

Based on the analytical models of Section (2.2.1) one could think that the network structure that best models the system will be close to the ones for modelling the dynamic evolution of the wheel wear. In fact, the network inputs and the size of the training dataset are the same. Therefore, based on the previous work (Section 5.2.1) and in order to demonstrate the power of the proposed methodology, in this case only three network structures are analysed. Actually, the network structures used for modelling the surface roughness are 5HN5D, 8HN8D and 10HN10D. Thus, the range of neurons in the hidden layer and delay units obtained after the coarse tuning for wheel wear (Section 5.2.1.1) is covered.

5.2.2.2 Analysis of Results

The MSE of each of the six trained nets for each net structure (HN-D) are shown in Table 10. The upper row (net1, net2 … net6) represents the trained network that corresponds to each initialization of the weights using the Widrow-Nguyen proposal as explained in Section 4.3.1.4. On the other hand, in the left column the network structures analysed are represented, i.e., the corresponding neurons in the hidden layer (HN) and delay units (D). It can be seen that the best (lowest) MSE results are yielded by the structure 8HN8D (0.0022 and 0.0036), coming up next 10HN10D structure (0.0037 and 0.0041). Finally, the worst results are achieved with the structure with less neurons in the hidden layer and less delays in the feedback (0.0040 and 0.0046). Moreover, analysing the results of the 8HN8D network structure, it can be observed that the errors yielded are quiet low (except the net3) compared to the other structures. Therefore, the best two nets correspond to the

structure 8HN8D followed by the 10HN10D. Finally, the 5HN5D structure nets have, overall, the highest MSE error.

	net1	*net2*	*net3*	*net4*	*net5*	*net6*
5HN5D	0.0111	0.0277	**0.0040**	**0.0046**	0.0859	0.0094
8HN8D	**0.0022**	**0.0036**	0.1457	0.0180	0.0057	0.0104
10HN10D	0.0203	0.0134	**0.0041**	0.0056	**0.0037**	0.1077

Table 10 Summary of the MSE results of the six trained nets for each ANN structure

In Table 11 the MAME (see Section 4.3.3.2) results for the best nets are summarized. It can be seen that the nets with the lowest MAME in units of process (µm) are the same that those with the lowest MSE. Thus, it can be concluded that the 8HN8D structure is the most suitable one to model the relationship between the specific grinding energy, wheel characteristics, operation characteristics and the surface roughness. However, although the selected networks have the lowest MSE and MAME, it is important to note that in the case of the net4 with structure 5HD5D, it has the highest MSE of the selected ones (Table 10) but the third lowest MAME. In fact, as in the case of the modelling of the wheel wear, it be can concluded that MSE is not enough to select the proper net.

	8HN8D net1	*8HN8D net2*	*10HN10D net5*	*5HN5D net3*	*10HN10D net3*	*5HN5D net4*
MSE	0.0022	0.0036	0.0037	0.0040	0.0041	0.0046
MAME	**0.26**	**0.32**	0.43	0.35	0.44	0.33

Table 11 Summary of the MAME errors of the best nets

Unlike for the tuning of the ANN-based sensor for wheel wear measuring, here only the two best networks are compared due to the less number of structures used.

In Figure 34 and Figure 35, the prediction for new grinding conditions for known grinding characteristics is shown. For the test experiment 1 (Figure 34) the behaviour of the two nets is quite similar. Both of them start at the same point and follow a similar tendency. However, around 480 mm^3/mm, the predicted signal of the network 8HN8D net2 increases while the target signal decreases. Thus, in the case of 8HN8D net2 the error is a bit higher. Regardless the initial error, the maximum error is 0.32µm. In the case of test experiment 2 (Figure 35), the behaviour of both nets is quite different. The worst performance corresponds to the 8HN8D net2. The maximum error for the 8HN8D net2 is 0.24µm, without taking into account the initial errors, while the error for the 8HN8D net1 is around 0.1µm. However, it can be seen that for the 8HN8D net1 after around the prediction point number 1600 mm^3/mm the error starts to increase while for the 8HN8D

net2, after around point 1400 mm^3/mm the error decreases, being the end close to zero. Furthermore, after 600 mm^3/mm the signal of the network 8HN8D net1 remains almost constant, i.e. it shows the same phenomenon observed in the case of wheel wear. In any case, it can be said that the errors in both cases, even for the 8HN8D net2, are low from the point of view of the grinding process.

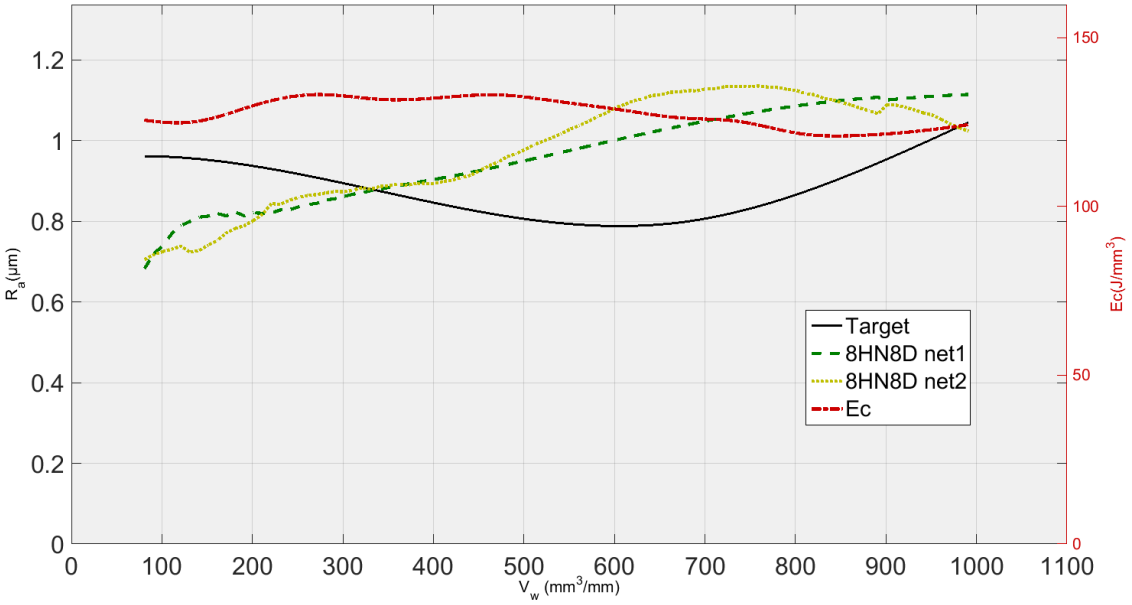


Figure 34 Surface roughness generalization capability of the two preselected networks. Test experiment 1: 82AA36K6VW q_s = 100; Q' = 2.5

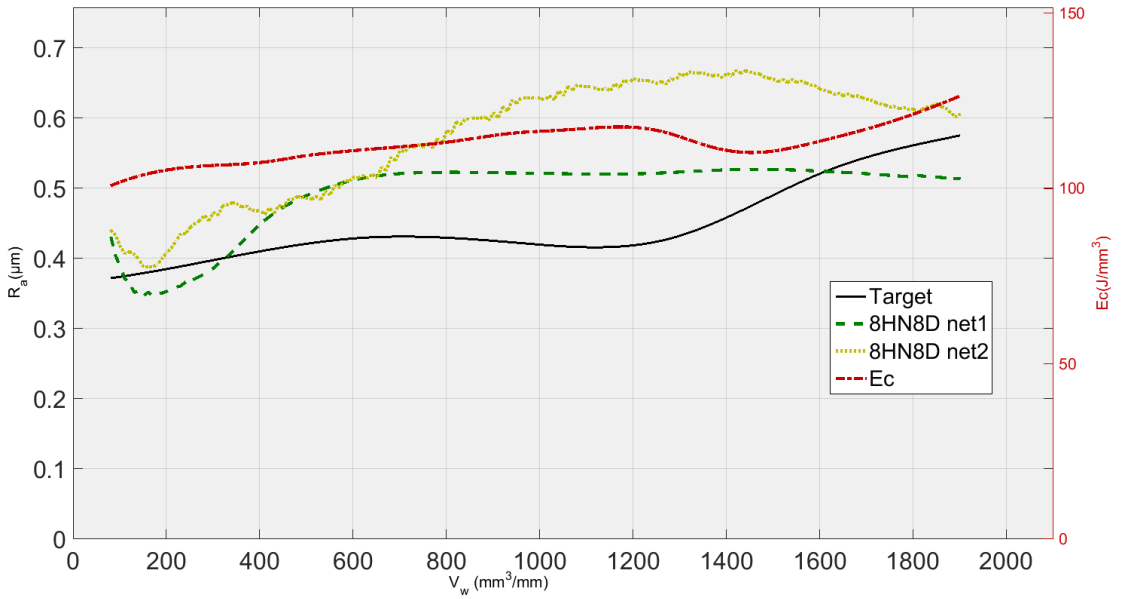


Figure 35 Surface roughness generalization capability of the two preselected networks. Test experiment 2: 82AA70G6VW q_s = 60; Q' = 1

It can be seen that for test experiment 3 and 4 (new wheel characteristics not used during the training process), the two nets predict quite well. In the case of test experiment 3 (Figure 36), it can be said that the net that more precisely predicts the surface roughness is the network 8HN8D net2, being the error in almost the complete dynamic evolution lower than 0.15µm, indeed, the highest error is around 0.16µm. Thus, the 8HN8D net2 trained ANN predicts up to 1800 mm^3/mm of the dynamic evolution of surface roughness with high accuracy. In the other net (8HN8D net1) it can be observed that the error is a bit higher but lower than 0.30µm. However, from 500mm^3/mm the signal remains constant. For the test experiment 4 (Figure 37), the results are a bit worst. At the beginning, both nets predict well but after 600 mm^3/mm, the error increases and the signal remains almost constant. In the case of the 8HN8D net 2 the prediction oscillates around one point, thus, it can be said that there is a quite similar phenomenon in both predictions. Likewise, it can be said that the error in both cases is quite similar, being a bit higher for the 8HN8D net2 (0.36µm).

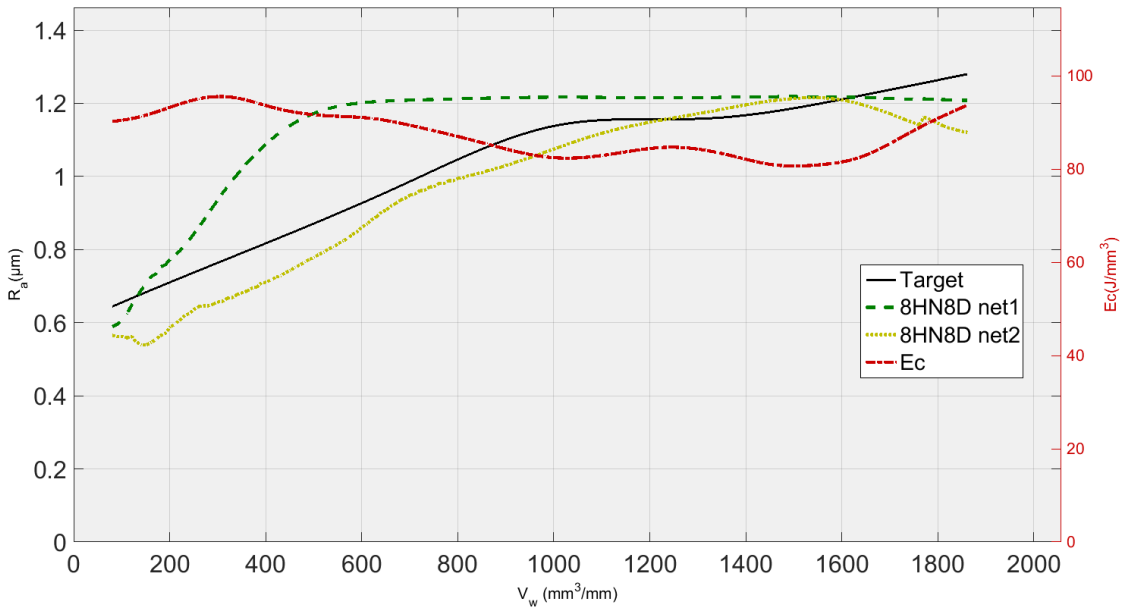


Figure 36 Surface roughness generalization capability of the two preselected networks. Test experiment 3: 82AA36G6VW q_s = 60; Q' = 2.5

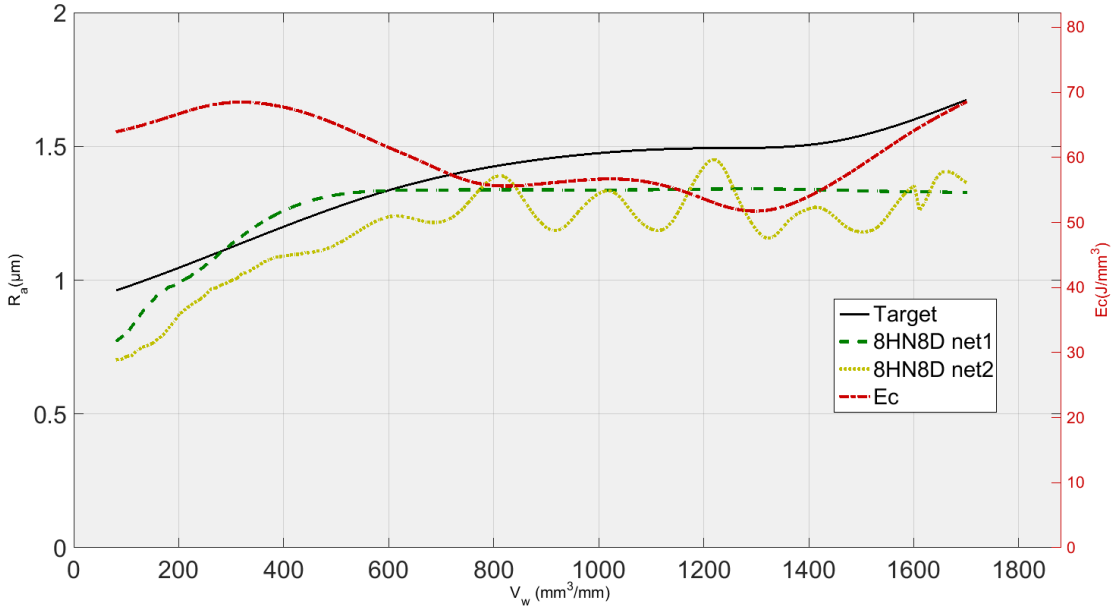


Figure 37 Surface roughness generalization capability of the two preselected networks. Test experiment 4: 82AA36G6VW q_s = 60; Q' = 4

Since the same phenomenon observed for wheel wear is repeated in the estimation of the surface roughness, it is appropriate to analyse how the nets predict the surface roughness in the final stage of the evolution. To that goal, the results of the CV of the final stage of the evolutions are shown in Table 12. It can be seen that for the net1 with 8 neurons in the hidden layer and 8 delay units (8HN 8D) the values of coefficient of variation are very low, especially for the test 2, test 3 and test 4. In the case of the 8HN 8D net2, the coefficient of variation is higher and, thus, the prediction does not remain constant after one point. At the sight of the results it can be said that for the second net with 8 neurons in the hidden layer and 8 delays in the feedback the constancy phenomenon is avoided, if not completely, at least it is to a high extent. To check this, in the next paragraphs the values of the coefficient of variation and the prediction performance are compared.

	Test	*8HN8D net1*	*8HN8D net2*
Coefficient of Variation (CV)	Test experiment 1	0.0133	0.0350
	Test experiment 2	0.0082	0.0299
	Test experiment 3	0.0029	0.0269
	Test experiment 4	0.0039	0.0726

Table 12 Results of the coefficient of variation analysis for the of the prediction horizon of surface roughness

In Figure 34 (test experiment 1) none of the two prediction signals remain constant after one point. However, the 8HN8D net1 prediction is barely an incrementally linear and this is the reason of the lower CV value, similar to the wheel wear case. The results about the

analysis of the coefficient of variation show that for the 8HN8D net1 in the test experiment 2, test experiment 3 and test experiment 4, at the end of the prediction horizon, the dynamic evolution remains almost constant. It clearly can be seen that the results of the Table 12 are coherent with Figure 35-10. As said before, the Figure 35 shows how the signal of the 8HN8D net1 remains constant after 600 mm^3/mm, as the coefficient of variation metric shows with a value lower than 0.01. In the case of the 8HN8D net2, the CV value is higher than for the 8HN8D net1 and, thus, it can be checked (see Figure 36) that the signal is not constant at the end. In the same way, the results of CV for the test experiment 4 show clearly that the output for the 8HN8D net1 is almost constant (see Figure 37).

In the view of the results, in this particular case (the prediction of the dynamic evolution of the surface roughness) where the prediction signals are not very oscillatory, the value of CV higher than 0.01 show that the prediction does not remain constant at the end. Thus, the networks that have yielded the lowest MAME with a coefficient of variation higher than 0.01 are considered the best networks for modelling the grinding surface roughness.

Moreover, it can be concluded that the coefficient of variation analysis is highly dependent on the application and modelling signal. Thus, the reference values to discern if a signal remains constant or not depends on the application and modelling signal and, consequently it is not possible to define a common reference CV value for all the cases. However, checking the results yielded for the wheel wear and surface roughness one could think that CV values lower than 0.01 are a good indicator of the presence of the constancy phenomenon.

To sum up, after examining the MSE, the MAME and the CV of the final part of the network output, the net that best models the dynamic evolution of the surface roughness avoiding the constancy of the signal is the 8HN8D net2. Although this net does not yield the lowest MSE and MAME error, it achieves the second lowest errors and in any of the test cases at the end of the signals the output does remain constant. Thus, the best net has 5 inputs, 8 neurons in the hidden layer, 8 delay units in the feedback and one output neuron.

5.3 Conclusions

In this Section a solution based on RNN to develop soft sensors to measure the wheel wear and surface roughness during the process with the ability to generalize to new wheel characteristics and grinding conditions has been presented.

The selected recurrent neural architecture, Layer-Recurrent Neural Network, has shown the potential for modelling the dynamic evolution of the wheel wear and surface roughness without measuring initial real values in a prediction horizon up to 2000 mm^2 of specific volume of part material removed. This is a remarkable task because it means that the proposed methodology is capable of predicting up to 200 points of a complete dynamic evolution without initial real values. In fact, the MAME error for wheel wear is 32 μm, and 0.26 μm for surface roughness.

The calibration process of the sensor involves establishing the best possible ANN structure (neurons in the hidden layer and delay units in the feedback) that can model with good accuracy the dynamic evolution of wheel wear or surface roughness. This is carried out in a two-stage process, which involves coarse and fine tuning of the sensor. In the coarse tuning the Mean Square Error (MSE) is used to select the best hidden neurons (HN) and delay units (D) range. Then, based on the range obtained from a coarse tuning, the fine tuning aims to obtain the best possible network structure.

However, it is not possible to select the best ANN structure only by comparing MSE values of the test dataset. Actually, the HU5D5 ANN structure for modelling wheel wear has achieved the lowest MSE value but it does not provide the best behaviour for modelling the dynamic evolution of wheel wear. Thus, two new *ad-hoc* indicators are proposed: the MAME value and the CV of the final stage of the prediction horizon. Those indicators help to select the best structure in a fine tuning calibration stage. However, the CV analysis is highly dependent on the application and modelling signal and, consequently, the reference values to discern if a signal remains constant or not depends on those characteristics.

The lowest errors (and therefore, the best performance of the sensor) are achieved with a neural configuration centred in 10 HN and 10 D for the wheel wear, and 8 HN and 8 D for the surface roughness. In other words, these network configurations satisfactorily represent the wheel wear and surface roughness. Thus, this result confirms that for modelling the dynamic evolutions of both wheel wear and surface roughness, the quantity

of hidden neurons and delay units is quite similar. Therefore, this confirms the initial hypothesis accepted by literature and expressed in Equations (1) and (2).

Finally, regarding the estimation of the evolution of wheel wear one might think that the characteristics of the wheel have a higher influence on the wear behaviour than the cutting parameters. In fact, the highest maximum error for wheels used and not used during the training process are 9 µm and 67 µm, respectively. Regarding the estimation of the surface finish, the wheel characteristics do not have such an influence on the surface finish behaviour. Actually, the selected net generalizes with good results to new wheels (not used during the grinding process), 0.36µm maximum error, and new grinding conditions, 0.32µm maximum error.

6 SPECIFIC GRINDING ENERGY MODELLING

Specific grinding energy is a fundamental variable in order to know the performance of the grinding process and it is also useful for estimating the power requirement of the grinding machine. Therefore, knowing beforehand the specific grinding energy from wheel characteristics and cutting conditions can help to know the performance of a grinding wheel in advance. The prediction of the specific grinding energy is done off-line, thus, before finding the best possible network, it is highly important to study the characteristics of the dynamic evolutions in order to select the most suitable ones for modelling the specific grinding energy. In the view of the results, 200 points are in this case enough to achieve satisfactory results. Thus, given a time horizon of 2000 mm^3/mm, the time step is set to 10 mm^3/mm. Considering the conclusions about the characteristics of the time series and the neural configuration, the new aim is to find an effective net configuration for predicting the specific grinding energy. The yielded MAME and relative errors are lower than 33.60 J/mm^3 and 23.65%, respectively. Although the proposed solution is able to predict specific grinding energy quite good, one unique RNN it is not able to predict the specific grinding energy for all the wheels commercially available. Thus, a neuro-fuzzy approach is proposed to generate custom networks for specific grinding wheels and grinding conditions using available experiments. Besides, the inputs of the fuzzy system are weighted with information extracted from a trained network in order "to help" the clustering. Under custom training datasets (weighted and non-weighted) and non-custom (all the experiments available), the custom networks obtained with the weighted approach yields slightly better results. Likewise, the results achieved with all the experiments and with custom datasets are similar. Finally, using custom training experimental time and money, and also, RNNs training time is saved.

As said in 2.2.2, little effort has been dedicated to the modelling of the dynamic evolution of the specific grinding energy with ANNs. However, it is a fundamental variable in order to know the performance of the grinding process and it is also useful for estimating the power requirement of the grinding machine.

Grinding wheels are produced in a semi-handmade process. Thus, it is not possible to know *a priori* relevant data about the performance of the wheel. Besides, production and delivery times are very long. Therefore, it is necessary to develop models to know precisely the behaviour of the wheel from its characteristics. The advantages of this approach are: avoiding bad buying decisions, avoiding very time-consuming experiments, and avoiding wasting of time due to the long delivery times. Thus, knowing beforehand the specific grinding energy from wheel characteristics and cutting conditions can help to know the performance of a grinding wheel in advance.

Therefore, to achieve the goal of modelling the dynamic behaviour of the specific grinding energy, the modelling of the complete dynamic evolution of the specific grinding energy is presented here. The developed model has to be capable to generalize for new wheels (not used during the training process) and grinding characteristics (not used for a specific wheel).

It is noteworthy that for modelling the specific grinding energy it is not necessary to introduce other inputs and, thus, the methodology explained in Chapter 4 is used. Given this methodology, this chapter contains the comparative study and the results.

6.1 Comparative study and results

In this work, the aim is to model the complete dynamic evolution of the specific grinding energy bounded in a specific prediction horizon, i.e. a finite quantity of points must be predicted. Thus, there are two characteristics that must be taken into account: time-step and prediction horizon. Time-step is defined as the gap between two consecutive points of the dynamic evolution (Figure 38a). On the other hand, prediction horizon is referred to the length of the complete dynamic evolution to be predicted (Figure 38b). These two characteristics define the quantity of points in the dynamic evolution.

In the case of the design of the soft sensors, the time-step is mostly given by the application requirements and the characteristics of the signal to be measured. However, the prediction of the specific grinding energy is made off-line or in advance, without

machining any part. Hence, before finding the best possible network, it is highly important to study the characteristics of the dynamic evolutions in order to select the most suitable ones for modelling the specific grinding energy.

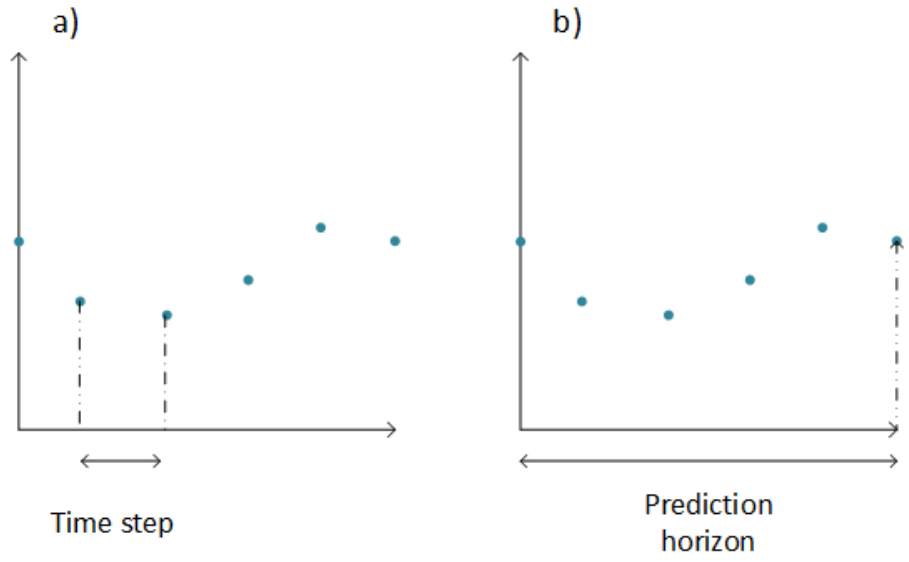


Figure 38 a) Time step. b) Prediction Horizon

On the other hand, as said before, the aim of the training process is to find the best possible number of neurons in the hidden layer (HN) and feedback units (D) for modelling the specific grinding energy. Therefore, in order to find the most suitable network structure for modelling the specific grinding energy, the trail-and-error approach is used as follows: first, coarse finding of the best HN and D range is carried out and, second, the fine finding is done in that HN-D range.

6.1.1 Dynamic evolution time characteristics vs. number of points

The influence of these two parameters (time step and prediction horizon) and the number of points is to be discussed in order to conclude their influence on the generalization capabilities of the ANN. In fact, the methodology of this study can be applied on further analysis on applications with similar characteristics. It should be noted that, in this work, the RNN acts as a model to be used off-line that provides a complete dynamic evolution in a specific and previously bounded time interval. Thus, certain flexibility can be afforded concerning the sizes of the time step and the prediction horizon as long as the generalization capability of the ANN is improved. Nevertheless, note that the length of the prediction horizon is dependent on the specific application and thus a significant reduction of the prediction horizon would decrease significantly the utility of the proposed RNN application.

The training process for this analysis is carried out varying the time step and the prediction horizon considering that the prediction horizon is to be slightly changed due to the application requirement. Thus, for analysing the influence of the time characteristics and the number of points three different net configurations (HN-D) have been selected (5HN 10D, 12HN 10D and 15HN 10D) due to the complexity of the task and experience gained developing the soft sensors (Chapter 4). In the selection process of the best network for the wheel wear and surface roughness the best results were yielded around 10 delays in the feedback. Besides, one could think that only with static inputs the net needs "more dynamic", which can obviously be addressed by adding feedback. These ANN configurations are trained ten times with time steps set to $5mm^3/mm$ (dataset 1, see Table 13) $10mm^3/mm$ (dataset 2, see Table 13) and $15mm^3/mm$ (dataset 3, see Table 13). Note that when the time step changes but the prediction horizon remains constant the number of points in the dynamic evolution also changes.

Besides, for analyzing the influence of the prediction horizon another training has been carried out with the reduction of the prediction horizon to the 2/3 while time step is set to $10mm^3/mm$ (dataset 4, see Table 13). The reduction ratio has been selected to ensure that the dynamic evolutions with this reduction have the same amount of points in each dynamic evolution as those with time step $15mm^3/mm$ and the prediction horizon 2000 mm^3/mm (dataset 3, see Table 13). The reduction has been done from each dynamic evolution and not from the global value ($2000mm^3/mm$) because some are shorter than others (see Section 4.3.2.1). Thus, it is possible to observe the influence of time step, prediction horizon and the number of points independently. In fact, one of the goals of this analysis is to compare the influence of the dynamic evolution characteristics (time step and prediction horizon) and the number of points on the generalization capabilities of the ANN. In Table 13 are collected the different datasets generated combining different time steps and prediction horizons. Thus, the characteristics of the dynamic evolutions that most influence on the prediction of the specific grinding energy are to be inferred as a conclusion of the analysis.

Datasets	*Time step*	*Maximum Prediction horizon*	*Number of points*
Dataset 1	5	2000	400
Dataset 2	10	2000	200
Dataset 3	15	2000	150
Dataset 4	10	1500	150

Table 13 Different combinations of time step and prediction horizon carried out

6.1.2 Network structure

After selecting the best time step and the maximum prediction horizon, in order to obtain the optimal number of hidden units in the hidden layer (HN) and delay units (D) for modelling the relationship between the network inputs and outputs, the training is divided in two phases, as explained in the general methodology:

1. *Coarse tuning:* Given the *coarse* nature of this phase and the complexity of the system, the configurations given by the combination of hidden neurons (HN) $\in$ {5, 10, 15, 20} and delay units (D) $\in$ {5, 10, 15, 20} are trained.

2. *Fine tuning:* the fine tuning of the training process is to be performed in the range given by the best structures inferred in the *coarse* tuning.

Each training of the RNN with a particular network structure is carried out ten times. For the initialization of the weights, the Nguyen-Widrow algorithm is applied.

6.1.3 Analysis of Results

First, the results of the analysis of the time characteristics are discussed and the best time step and prediction horizon are chosen. Finally, based on the previous analysis, the best network structure (neurons in the hidden layer and delays) for modelling the non-linear relationship between the wheel characteristics, grinding conditions and specific grinding energy is selected.

6.1.3.1 Time characteristics vs. number of points analysis

The aim of the following analysis is to provide clues about the influence of the dynamic evolution characteristics and ANN configurations on the generalization capabilities in the application scope previously defined.

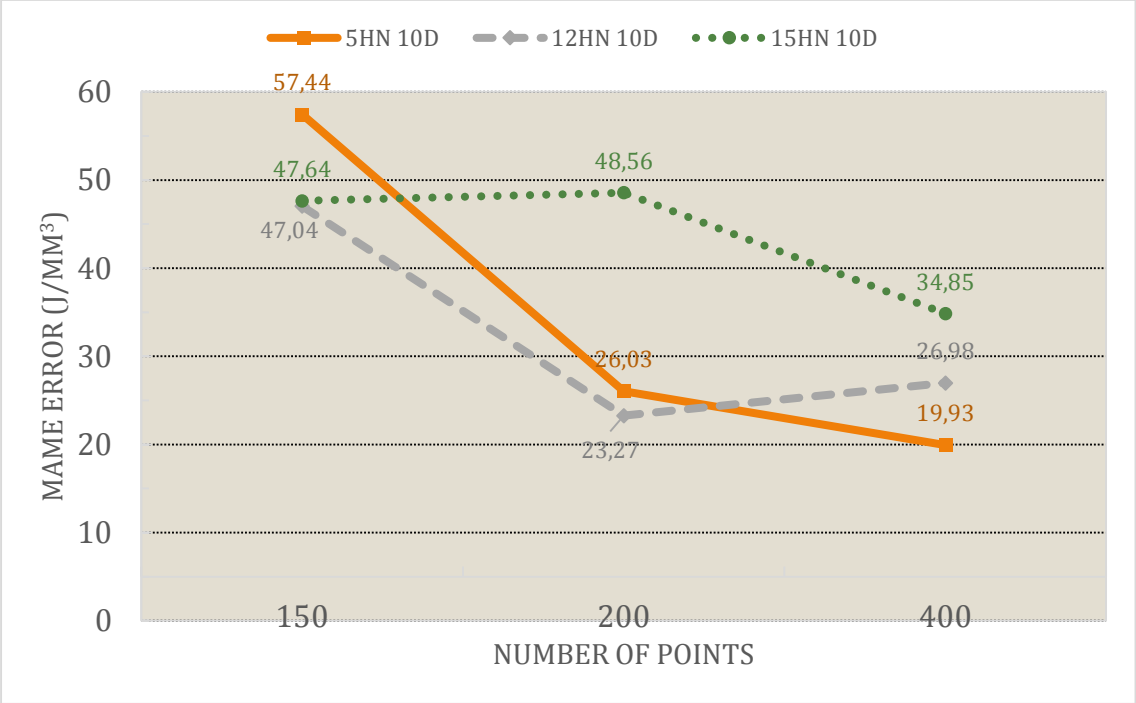


Figure 39 Analysis of the number of points of the dynamic evolution with constant time horizon (2000mm^3/mm)

Figure 39 shows the best MAME results yielded with the dynamic evolution with different number of points generated through changing time step (Dataset 1, Dataset 2 and Dataset 3) and different ANN topologies. The minimum MAME results are yielded for dynamic evolutions with more than 200 points (Dataset 1 and 2). Moreover, 15HN 10D configuration yields unsatisfactory results independently of the number of points. Thus, it is considered that this configuration is not suitable for modelling the application process and that is why it is discarded. Therefore, with the exception of 15HN 10D configuration, it can be noticed that from 200 points lower errors are obtained and, compared with the results obtained with 200 points, no significant improvement is shown with 400 points. Furthermore, comparing the Dataset 2 and the Dataset 4 (Figure 40), this latter generated with the same time step of 10mm^3/mm but with less points (a shorter time horizon), it can be observed that the best results are obtained when using more points. Thus, with the available results it can be concluded that, in this kind of ANN applications, nets trained with dynamic evolutions with more points generalize better.

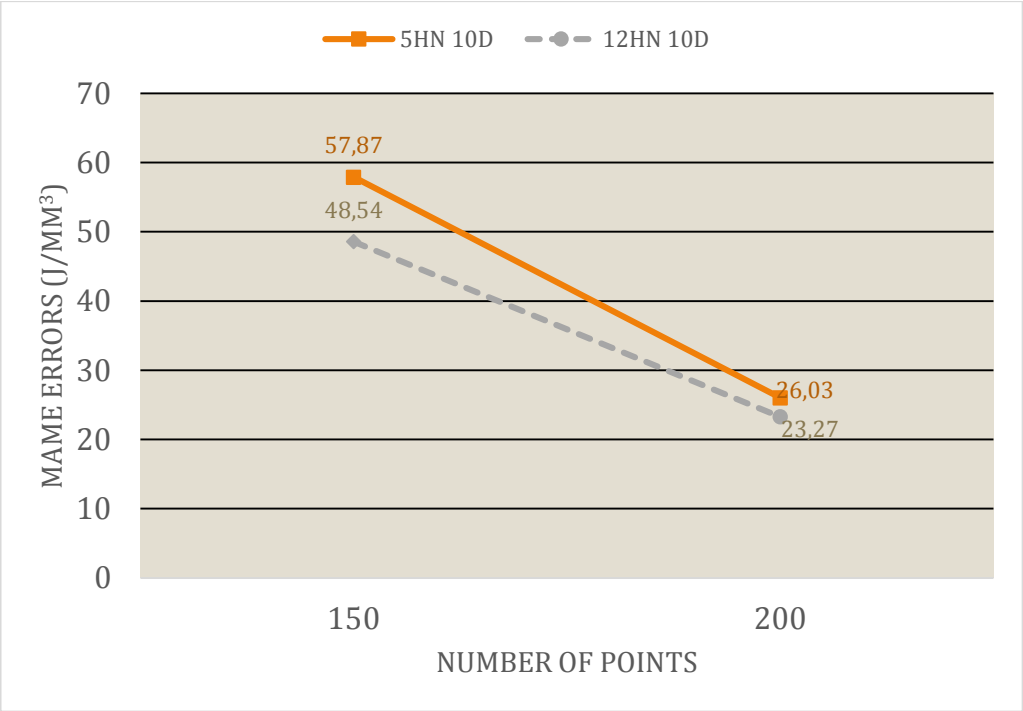


Figure 40 Dataset 2 and Dataset 4 analysis with constant time step (10mm^3/mm)

However, not only the number of points in the dynamic evolution can affect the generalization capabilities of the net. It can be observed (Figure 39) that the results with Dataset 2 (200 points) are slightly better than the achieved for the Dataset 3 (400 points). It might be due to the excessive resolution of the dynamic evolution with the time step of 5. Thus, in dynamic evolutions a too short time step might increase the prediction error with the additional disadvantage of also increasing the training time. This means that nets trained with evolutions with lower time step do not ensure better generalization. To reinforce the previous conclusions, Figure 41 shows that the net trained with evolutions with wider time step (Dataset 3) yields similar results, 47.04 J/mm^3, than the net trained with the same number of points but narrow time step (Dataset 4), 48.54 J/mm^3.

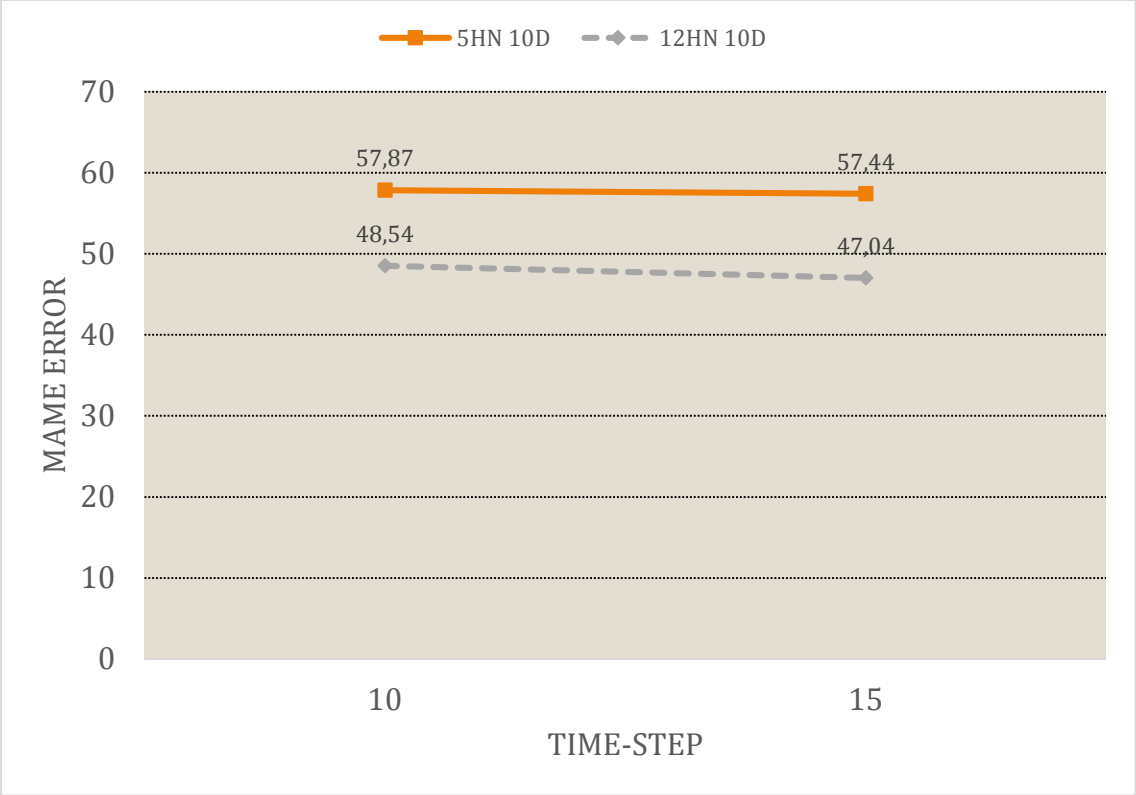


Figure 41 Time step and time horizon analysis with constant number of points

In summary, it can be concluded that when the dynamic evolutions database for the ANN training is generated, the critical point to be considered is the number of points. Depending on the task to solve with the ANN, an analysis of the time step of the dynamic evolution has to be done to improve the generalization capabilities of the net. In this particular case, 200 points are enough to achieve satisfactory results. Given a time horizon of 2000mm^3/mm, the time step is set to 10mm^3/mm. On the other hand, the analysis of the influence of the time characteristics and the number of points shows that the best results are yielded within the range from 5 to 12 hidden units (HN). In fact, the net performs better with less than 12 neurons in the hidden layer.

6.1.3.2 Best network structure selection

Considering the conclusions of the previous section about the characteristics of the time series and the neural configuration, the new aim is to find an effective net configuration for predicting the specific grinding energy. Thus, the following ANN training process is carried out with time series with up to 200 points.

Figure 42 gathers the best ten nets (the ten nets with the lowest MSE error) and the corresponding neurons in the hidden layer and feedback delays (HN-D) configuration showing how the performance (MSE) varies for different HN-D configurations. It can be

observed that the brown point corresponding to the 5HN15D configuration yields the highest error. Moreover, from all the trained nets only one of the best ten nets has 5 neurons in the hidden layer. In addition, 4 of the ten nets correspond to 7 neurons in the hidden layer. One of these, the light-blue one with 14 delays, yields good results, an MSE error below 0.0030. However, 5 of the best ones have 10 neurons in the hidden layer. Two of them (green and blue) yield the same lowest MSE error with 12 delays and 5 delays, respectively, and the one represented with the dark-blue colour provides an MSE error slightly higher than 0.0030. It can be observed that the results of the net varies for different neurons in the hidden layer and delays of the feedback. In view of the results obtained in Figure 9, it can concluded that better results are achieved with more neurons in the hidden layer (note that from 10 neurons onwards higher MSE values are yielded due to overfitting). Moreover, as said in the Section 4.2.1 with a large quantity of neurons in the hidden layer the training time consumption increases considerably. Likewise, it can be noticed that the more neurons in the hidden layer, the lower the number of delays in the best nets. Actually, for 5HN the only best net has 15 delays. For 7HN, the number of delays of the best nets ranges from 14 to 18 delays and, finally, all the best nets obtained with 10HN have less than 13 delays in the feedback.

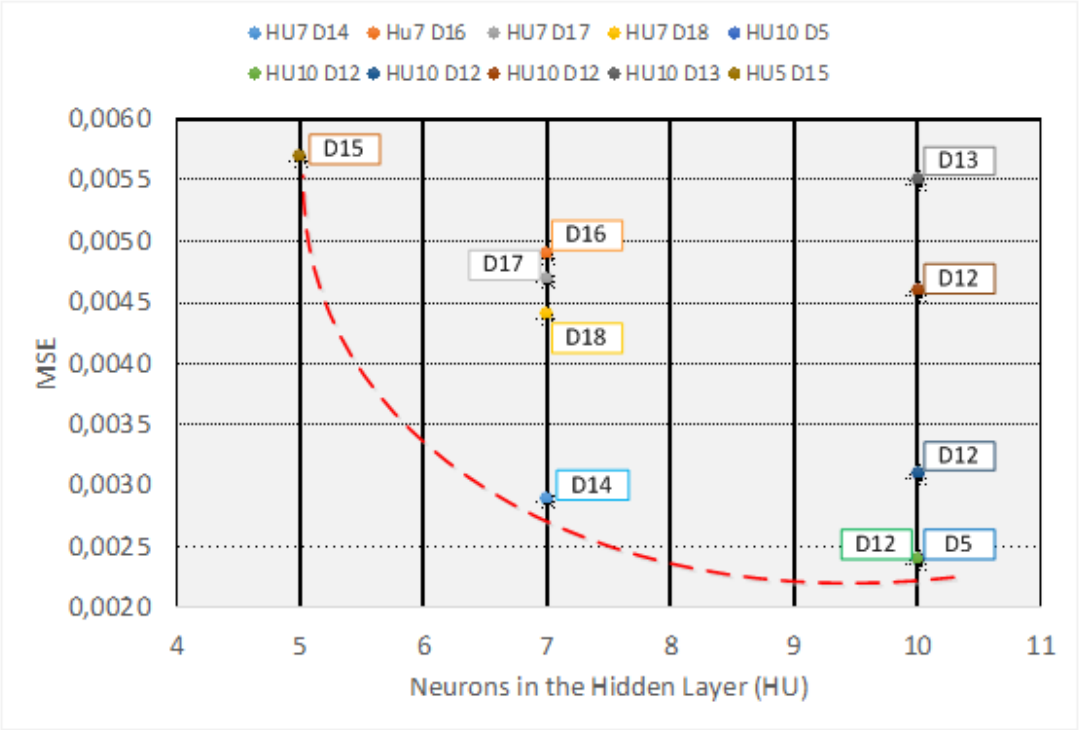


Figure 42 MSE results for different HN-D configurations (200 points)

On the other hand, Table 14 shows the numerical data of the ten nets represented in Figure 42. The number added to the word 'net' on Table 14 denotes the network number among the ten trained for each HN-D configuration. It can be observed that the best results have been yielded with more than seven neurons in the hidden layer. Besides, except two cases, the best results correspond to nets with a feedback delay around 13. However, it should be noted that there are excellent results with one network with 5 delays in the feedback. From the results shown in Table 14 it can be concluded that most of the best MSE results are yielded around within the range 7-10 neurons in the hidden layer and 13 delays. However, as explained in Chapter 3, the MSE results are not enough to select the best network from the application point of view and, consequently, a further study has to be conducted. Thus, the MAME error is calculated for the nets with the lowest MSE error (in bold).

Hidden Units (HN)	Delays (D)	Net	MSE
7	**14**	**Net3**	**0.0029**
7	16	Net6	0.0049
7	17	Net9	0.0047
7	**18**	**Net8**	**0.0044**
10	**5**	**Net8**	**0.0024**
10	**12**	**Net5**	**0.0024**
10	**12**	**Net6**	**0.0031**
10	12	Net10	0.0046
10	13	Net6	0.0055
5	15	Net1	0.0057

Table 14 MSE results of the best nets (200 points)

In Table 15 MAME error results for the nets with the lowest MSE error are shown. It can be observed that the best results are yielded with 10 neurons in the hidden layer and 12 delays in the feedback with a MAME error of 16.91 J/mm^3. This network configuration (10HN 12D) has a high repeatability because two nets have MAME errors lower than 20 and three networks with this configuration can be found in Table 14 with MSE errors lower than 0.0046. Besides, the nets with the lowest MAME error have around 13 delays in the feedback. Actually, the worst results is yielded with 10 neurons in the hidden layer and 5 delays despite of having the lowest MSE error.

	7HN14D net3	*7HN18D net8*	*10HN5D net8*	*10HN12D net5*	*10HN12D net6*
MSE	0.0029	0.0044	0.0024	0.0024	0.0031
MAME (J/mm^3)	**17.77**	22.89	26.98	**16.91**	19.43

Table 15 Best nets with the MAME result (200 points)

The predictions for new grinding conditions for known grinding characteristics are shown in Figure 43 and Figure 44. For the test experiment 1 (Figure 43) the behaviour of the two nets is different. Both of them start at the same point and follow a similar tendency. However, around 500 mm^3/mm, the predicted signal of the network 7HN14D net3 starts to fluctuate. On the other hand, in the case of 10HN12D net5, after 300 mm^3/mm start linearly increasing to a maximum error of 33.60 J/mm^3 at the end of the signal. In the case of the second test experiment (Figure 44), the behaviour of both nets is almost the same. However, at the end, the 10HN12D net5 prediction increases slightly and so as the maximum error. Nevertheless, the maximum errors are similar, 28.60 J/mm^3 for the 10HN12D net5, and 23.07 J/mm^3 for the 7HN14D net3. Although the maximum error of both nets is higher than the 10% limits, before 1200 mm^3/mm both predictions follows closely the target signal.

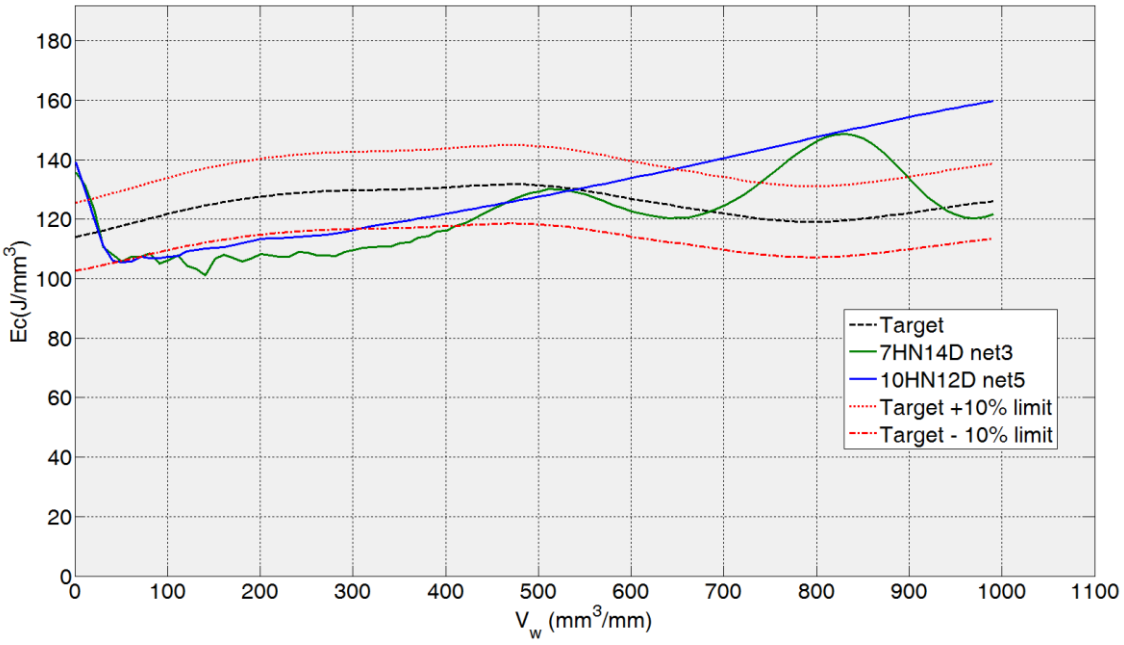


Figure 43 Specific grinding energy generalization capability of the two preselected networks. Test experiment 1: 82AA36K6VW q_s = 100; Q' = 2.5

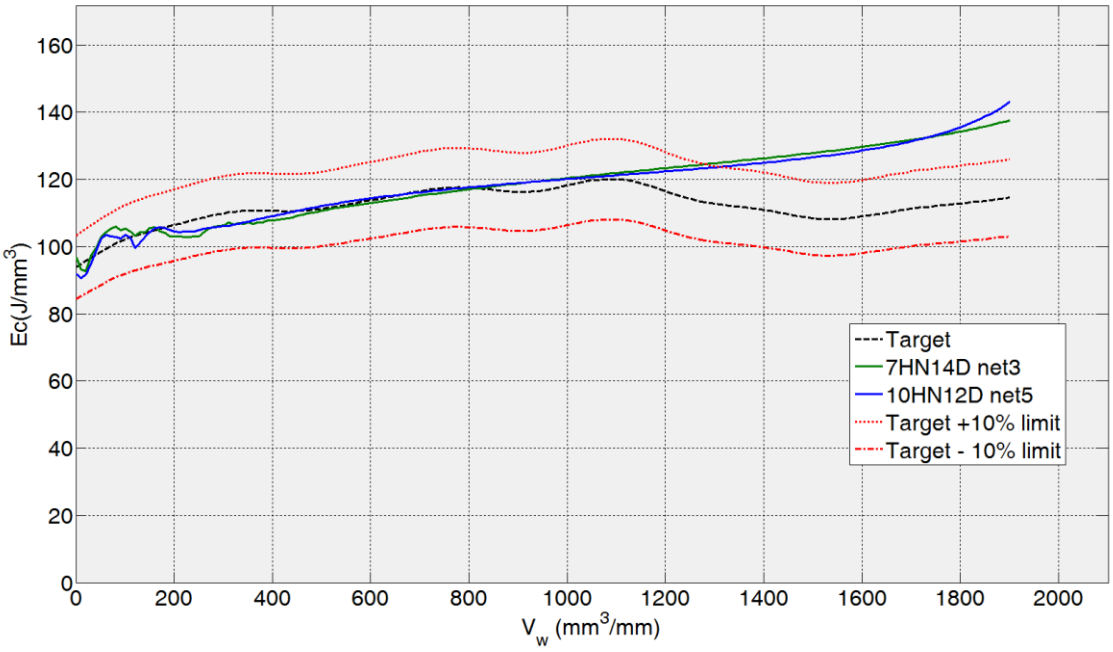


Figure 44 Specific grinding energy generalization capability of the two preselected networks. Test experiment 2: 82AA70G6VW q_s = 60; Q' = 1

Similar to the test experiment 2, test experiment 3 and 4 (new wheel characteristics not used during the training process), the two nets predict quite similarly. In the case of test experiment 3 (Figure 45), it can be said that the net that better predicts the specific grinding energy is the network 10HN12D net5, being the error in the complete dynamic evolution lower than around 10 J/mm^3. Besides, the prediction remains most of the time within the ±10% range. Nevertheless, the 7HN14D net3 trained ANN maximum error is a bit higher (15.94 J/mm^3) and from around 300 mm^3/mm to 780 mm^3/mm the prediction specific grinding energy is out of the ±10% limits. For the test experiment 4 (Figure 46), the results are similar. Although the maximum error of the 10HN12D net5 is higher (13.73 J/mm^3) than of the 7HN14D net3 (11.35 J/mm^3), this occurs at the end of the prediction. However, analysing the whole prediction horizon, it can be seen that the prediction of the 10HN12D net5 keeps within the ±10% range during more time. At any rate, both nets predictions are quite accurate for test experiments 3 and 4, with maximum errors lower than 16 J/mm^3. Likewise, the error is almost during all the prediction window lower than 10%, being slightly higher only at the end.

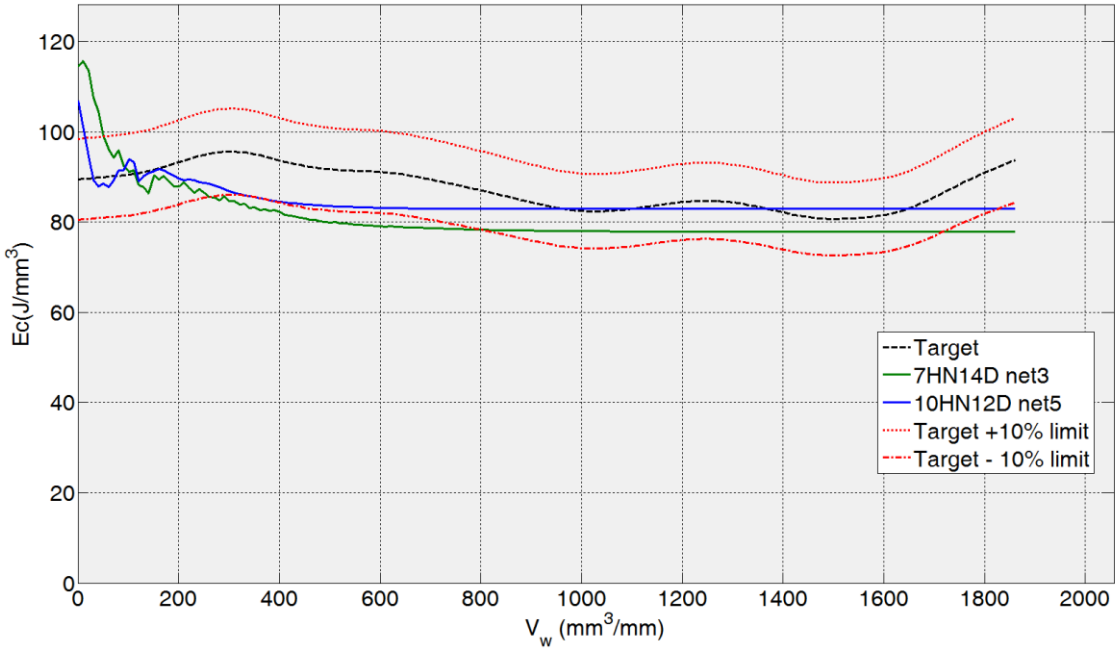


Figure 45 Specific grinding energy generalization capability of the two preselected networks. Test experiment 3: 82AA36G6VW q_s = 60; Q' = 2.5

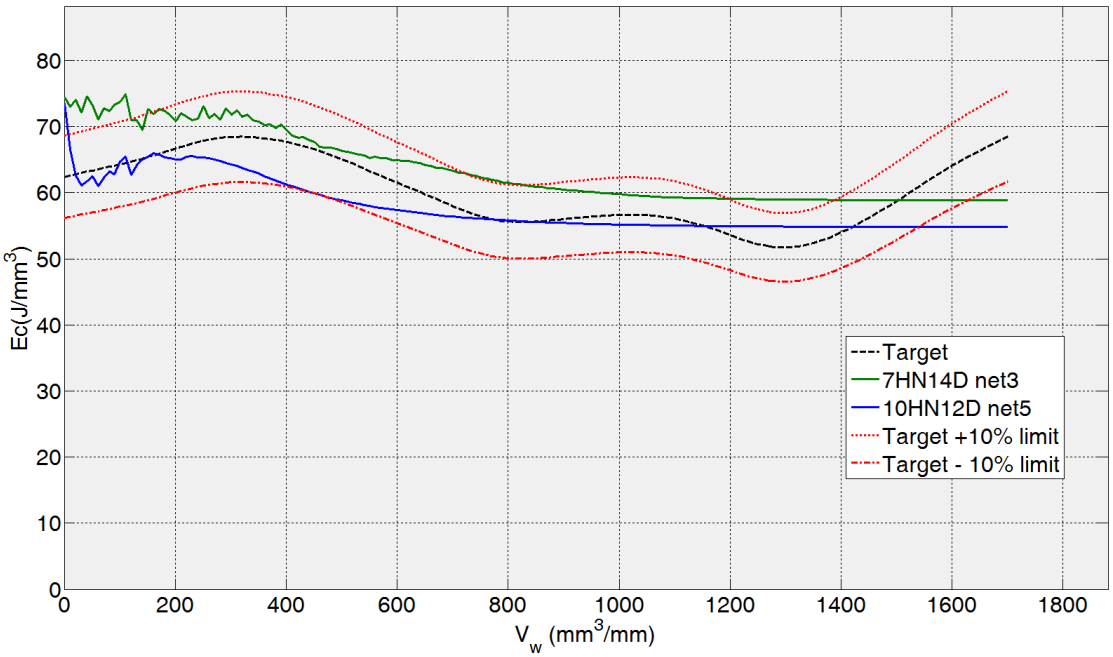


Figure 46 Specific grinding energy generalization capability of the two preselected networks. Test experiment 4: 82AA36G6VW q_s = 60; Q' = 4

Regarding the constancy phenomenon observed in the virtual sensors, in this case it is not so evident. In principle, for the same test experiment the training process has not yielded "good nets" some of them leading to constancy and others not leading to constancy. However, the behaviour of the best nets for test experiments number 3 and 4 could be

associated to the constancy phenomenon, and actually, compared to the virtual sensors, given the more complex nature of this estimation paradigm (with all the inputs static), one could think that the same phenomenon observed for wheel wear and surface roughness is repeated in the estimation of the specific grinding energy. Therefore, it is appropriate to analyse how the nets predict the specific grinding energy in the final stage of the dynamic evolution. To that goal, the coefficient of variation CV of the final stage of the evolutions is shown in Table 12. It can be seen that for test experiments 3 and 4 the CV is very low and, thus, the predictions remain constant after one point. On the other hand, for test experiments 1 and 2 the CV values are higher and the prediction does not remain constant at the final stage of the prediction. To check this, in the next paragraphs the values of the coefficient of variation and the prediction performance are compared.

	Test	*7HN14D net3*	*10HN12D net5*
Coefficient of Variation (CV)	Test experiment 1	0.0841	0.0260
	Test experiment 2	0.0213	0.0351
	Test experiment 3	6.47e-05	3.45e-05
	Test experiment 4	9.86e-05	3.16e-04

Table 16 Results of the coefficient of variation analysis for the of the prediction horizon of specific grinding energy

In Figure 43 (test experiment 1) none of the two prediction signals remain constant after one point. However, the 10HN12D net5 prediction is barely an incrementally linear while the 7HN14D net3 prediction is oscillatory. This is the reason of the lower CV value for 10HN12D net5. In the case of test experiment 2, the CV values are quite similar. The bigger CV of the 10HN12D net5 is because at the end the prediction increases slightly. For the third test experiment and fourth test experiment, at the end of the prediction horizon, the dynamic evolution remains almost constant for both nets. It clearly can be seen that the results of the Table 12 are coherent with Figure 43-46. As said before, the Figure 45 shows how the signal of both nets remains constant after 600 mm^2, as the coefficient of variation metric shows with a value lower than 0.01. Likewise, for test experiments 4, both predictions remain constant after 800 mm^3/mm.

In the view of the results, in this particular case (the prediction of the dynamic evolution of the specific grinding energy) the value of CV higher than 0.01 shows that the prediction does not remain constant at the end, as for the surface roughness, indeed.

In conclusion, after examining the MSE, the MAME and the CV, the net that best predicts the dynamic evolution of the specific grinding energy is the 10HN12D net5. In fact, this

net yields the lowest MSE and MAME error. Thus, the best net has 4 inputs, 10 neurons in the hidden layer, 12 delays in the feedback and one output neuron.

6.2 Developing custom networks with a neuro-fuzzy approach

It is evident that one unique RNN cannot predict the specific grinding energy for all the wheels commercially available. Furthermore, this statement can be applied to many application problems, obviously including the virtual sensors addressed in this work. Thus, this section deals with the proposal and methodology for generation of custom networks for specific grinding wheels and grinding conditions using available experiments. In particular, this section focuses on developing custom networks for the estimation of the specific grinding energy so as in future the methodology proposed by this work can be applied to the estimation of other grinding variables.

Two factors have direct influence on the performance of the network under supervised learning. First, the ANN structure and complexity. Second, the representational accuracy of the data used to train the net (Philip, 2009). The ANN structure and complexity were solved with the general methodology presented in Chapter 4. Therefore, the methodology for representational accuracy of the dataset used to train custom nets is presented in this Section.

Actually, selecting the proper training data has a great influence on the performance of the net. Besides, in grinding, performing grinding experiments is a highly time and resource consuming task. Therefore, cutting down on the number of experiments and selecting the custom training dataset is highly recommended and a desirable step forward in order to generate custom models for grinding process variables. However, downsizing too much the database may bring the reduction of the accuracy of the ANN. Some related works have analysed the effect of the training datasets over the network performance (Zhou & Wu, 2011) (Anjos, et al., 2015). These works showed that the training datasets have direct and decisive influence over the supervised learning of artificial neural networks. Thus, it is highly important to process training datasets for the improvement of the training performance. Other works have used different techniques in order to select custom training datasets. In (Ren, et al., 2014), it was proposed and compared the lateral data selection and longitudinal data selection for time series prediction. The results showed that by selecting specific training datasets from the whole available database the results improved. However, that selection looks more likely for forecasting in order to

select the best time windows for predicting future events. Furthermore, the k-means have also been used to downsize the dataset (Faraoun & Boukelif, 2007) (Young II, et al., 2008) (Zhuo, et al., 2014). In other cases, the selection is performed by fuzzy c-means (He, et al., 2005). Fuzzy c-means and k-means are quite similar. However, k-means is hard clustering. In hard clustering the data is divided into distinct groups, ergo, each datum only belongs to one cluster, which can be a rigid approach for those application fields without clear boundaries between categories, as is the case of grinding. Conversely, in fuzzy clustering, each datum can belong to more than one cluster with a membership level (Hicham, et al., 2012).

Consequently, in this work, the fuzzy clustering is used for selecting the custom training dataset. Within different fuzzy clustering, in this case, fuzzy c-means (FCM) are used because, compared to the other traditional clustering methods, fuzzy c-means are able to deal with the ambiguity and uncertainty of nature more closely (Zhou, et al., 2013). Besides, an advantage over other clustering methods is that it may be used in applications where the groups are overlapping (Pimentel & de Souza, 2013). FCM are a popular and widely used clustering technique. In (Khalid, et al., 2014) fuzzy c-means are used for segmenting the optic cup and optic disc for the CDR. They have been also used for clustering of coal seams based on their tendency to spontaneous combustion (Shau, et al., 2012). In (Nanda, et al., 2010) the FCM are used to group Indian stock market into clusters. Fuzzy methods have been also used in grinding for surface roughness prediction of ground components (Ali & Zhang, 1999), as well as for predicting surface finish and power requirement as part of a genetic-fuzzy system (Nandi & Pratihar, 2004).

However, none of the works found in the literature review have proposed any strategy for custom modelling of grinding process variables by means of a reduced and explicitly delimited dataset. Since grinding operations cover a very large number of components and requirements, one unique trained neural network would not be able to model all these operations. Actually, a new strategy must be developed to generate custom ANNs using the existing database or generating a new one at the lowest resources and time cost. Besides, downsizing the dataset means reducing the time and costs that involve carrying out the experiments to generate the database.

Therefore, in the methodology presented in this Section, a neuro-fuzzy system is proposed for predicting the dynamic evolution of the specific grinding energy for custom grinding wheels and grinding conditions. Actually, the fuzzy c-means are used to generate ad hoc

networks for specific grinding operations, involving, also, the reduction of the ANN training database and therefore, the amount of grinding experiments to carry out.

By means of FCM clustering, it is not possible to know in advance which grinding conditions and wheels fit better each other. However, it is possible to extract from previously trained networks (see Section 6.1.3.2) the network inputs that have bigger influence over the estimation of the specific grinding energy. Therefore, unlike other approaches for reducing the dataset using fuzzy c-means, in this work the inputs are weighted in order to "help" the FCM by extracting knowledge from the weights of previously trained ANNs, and thus "illuminating" the black box.

6.2.1 Neuro-fuzzy approach

In order to achieve the objectives a neuro-fuzzy system is proposed. To that end, first, the fuzzy part is used to downsize and select the custom training database and, second, the neuro part is used to model the specific grinding energy using the previously downsized training database as explained in Chapter 4 (Figure 47).

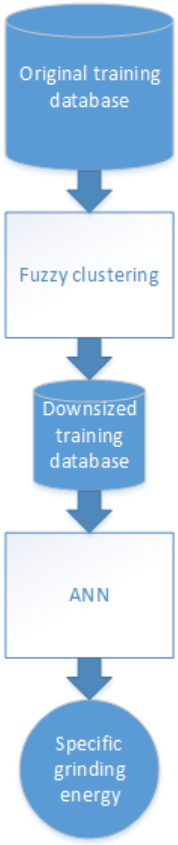


Figure 47 Neuro-fuzzy system block diagram

6.2.2 Original experimental database

The original database used in the neuro-fuzzy approach is the same as the one used to develop ANNs for predicting the specific grinding energy as described in Chapter 4. Likewise, the database has four features: grit size, hardness, Material removal rate (Q') and Speed ratio (q_s). Based on these inputs, in Table 17 the original database is shown.

Exp.	Grit size	Hardness	q_s	Q'	*Exp.*	Grit size	Hardness	q_s	Q'
1	36	0,24	60	1	*25*	100	0,24	60	4
2	36	0,24	60	2,5	*26*	100	0,24	80	1
3	36	0,24	60	4	*27*	100	0,24	80	2,5
4	36	0,24	80	1	*28*	100	0,24	80	4
5	36	0,4	60	1	*29*	100	0,24	100	1
6	36	0,4	60	2,5	*30*	100	0,24	100	2,5
7	36	0,4	60	4	*31*	100	0,36	60	1
8	36	0,4	80	1	*32*	100	0,36	60	2,5
9	36	0,4	80	2,5	*33*	100	0,36	60	4
10	36	0,4	80	4	*34*	100	0,36	80	1
11	36	0,4	100	1	*35*	100	0,36	80	2,5
12	36	0,4	100	2,5	*36*	100	0,36	80	4
13	36	0,4	100	4	*37*	100	0,36	100	1
14	70	0,24	60	1	*38*	100	0,36	100	2,5
15	70	0,24	60	2,5	*39*	100	0,36	100	4
16	70	0,24	60	4	*40*	70	0,36	60	1
17	70	0,24	80	1	*41*	70	0,36	60	2,5
18	70	0,24	80	2,5	*42*	70	0,36	60	4
19	70	0,24	80	4	*43*	70	0,36	80	1
20	70	0,24	100	1	*44*	70	0,36	80	2,5
21	70	0,24	100	2,5	*45*	70	0,36	80	4
22	70	0,24	100	4	*46*	70	0,36	100	1
23	100	0,24	60	1	*47*	70	0,36	100	2,5
24	100	0,24	60	2,5	*48*	70	0,36	100	4

Table 17 Original experimental database

Therefore, the aim of the fuzzy phase is to select experiments using fuzzy clustering from the original database in order to generate custom networks for specific grinding wheel and grinding conditions.

6.2.3 Fuzzy clustering to select the custom database

Clustering is related to the partitioning of a dataset into subsets or clusters. Thus, the data in each cluster shares some common features (Bataineh, et al., 2011). Within different clustering methods Fuzzy c-means (FCM) is the most used one. FCM is an unsupervised clustering algorithm proposed firstly by Dunn (Dunn, 1973) and later improved by Bezdek (Bezdek, 1981). The FCM try to find the most characteristic point of each cluster

and the grade of membership of each datum to the clusters (Wang & Zhang, 2007). For this, the objective cost function is minimized:

$$J_m(U,V,X) = \sum_{j=1}^{n}\sum_{i=1}^{c} u_{ij}^{m}\left\|x_j - v_i\right\|^2 \quad (16)$$

where n is the number of patterns of the dataset; c is the number of clusters; $U = [u_{ij}]_{cxn}$ is a fuzzy partition matrix; $X = \{x_1, x_2, \ldots, x_n\} \subset R^s$ is a vector of feature data; $V = \{v_1, v_2, \ldots, v_c\} \subset R^s$ is the vector of cluster centres and $\left\|x_j - v_i\right\|$ is the Euclidean norm between x_j and v_i. m is the parameter that fixes the fuzziness of the resultant clusters. The necessary conditions for minimizing the objective cost equation are:

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{\left\|x_j - v_i\right\|^2}{\left\|x_j - v_k\right\|^2}\right)^{\frac{1}{(m-1)}}} \; 1 \le i \le c, 1 \le j \le n \quad (17)$$

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} x_j}{\sum_{j=1}^{n} u_{ij}^{m}} \; 1 \le i \le c \quad (18)$$

The number of clusters c, the fuzziness exponent m, the tolerance ε, and, of course, the vector of features have to be introduced before carrying out the FCM algorithm. Thus, it is important to highlight that the number of clusters have to be introduced before using the FCM algorithm. Usually, FCM algorithm is used with different numbers of clusters, and once the procedure is finished the best suited cluster number is selected by validity indices.

The feature matrix is usually normalized within the range [0, 1]. However, one could think that the knowledge about the process can be used to weight the inputs in order to obtain more realistic clustering. Therefore, in this work the knowledge about the grinding process deduced from the best ANN in Section 6.1.3.2 is 'illuminated'. Actually, this trained ANN, which is able to predict the specific grinding energy with high accuracy but with a "big database", provides important information about the influence of the inputs on specific grinding energy that can be helpful to cluster that big database.

Actually, it is said that the ANNs are ''black box's''. Indeed, this is a major weakness compared to traditional analytical approaches that can easily quantify the influence of each independent variable on the modelling process (Olden & Jackson, 2002). However, some works have tried to determine the importance of the inputs parameters over the outputs of the ANN (Paliwal & Kumar, 2011) (Kemp, et al., 2007). In this case, the

connection weight method proposed by Olden and Jackson is one of the well-recognized (Olden & Jackson, 2002). The connection weight method sums the product of the weights from the input neurons to the hidden neurons with the weights from the hidden neurons to the output neurons for each input parameter. The larger the sum of weights, the greater the importance of the evaluated input parameter. Thus, the relative importance of each input parameter is determined by the following equation:

$$Imp(i) = \sum_{x=1}^{n}(CW_{ih(x)}CW_{ho(x)}) \quad (19)$$

where *Imp(i)* is the relative importance of the input parameter *i*; n the total number of hidden neurons in the hidden layer; x the index number of the hidden neuron; $CW_{ih(x)}$ the weight between the input parameter *i* and the hidden neuron *x*; and $CW_{ho(x)}$ the weight between the hidden neuron *x* and the output neuron.

As said before, once the FCM algorithm is used, some kind of validation indices are needed in order to select the best number of clusters. When the data is two-dimensional is easy to visualize the solution and verify the best number of clusters. However, for more than two dimensions the visual verification is not possible, thus, new validation indices are required. Wang and Zhang studied the number of fuzzy cluster validity indices in the literature and concluded that none of these indices correctly recognizes the optimal cluster number (Wang & Zhang, 2007). Therefore, in most of the cases the optimal number of clusters are decided by applying more than one index. The common indices used to select the most suitable number of clusters are the following ones (Balasko, et al., 2005):

- Partition Coefficient (PC): it measures the amount of "overlapping" between clusters (Bezdek, 1981).
- Classification Entropy (CE): measures the fuzzyness of the cluster partition only. It is quite similar to the Partition Coefficient index.
- Partition Index (SC): it measures the ratio of the sum of compactness and separation of the clusters (Zahid, et al., 1999).
- Separation Index (S): in contrast to partition index (SC), the separation index uses a minimum-distance separation.
- Xie and Beni index (XB): it quantifies the ratio of the total variation within clusters and the separation of clusters.
- Dunn's Index (DI): this index identifies "compact and well separated clusters".
- Alternative Dunn Index (ADI): it is a modification of the original Dunn's for simpler calculation.

However, although Partition Coefficient (PC) can be used as a validity index, they are irrelevant to the problem of cluster validity (Trauwaert, 1988). Likewise, Classification Entropy index and the PC are quite similar. On the other hand, the Dunn's Index (DI) is high resource consuming. Besides, the DI index and Alternative Dunn Index (ADI) indices are not very reliable for fuzzy clustering because of re-partitioning the results with the hard partition method (Balasko, et al., 2005). Therefore, in this work the Partition Index (SC), the Separation Index (S) and the Xie and Beni index (XB) are used to select the optimal number of clusters.

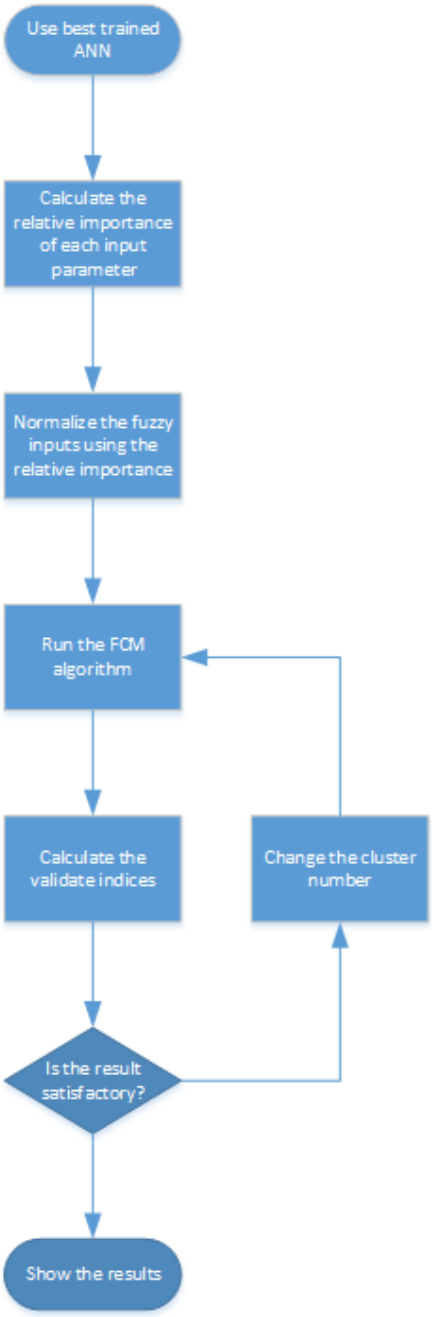


Figure 48 Fuzzy part flowchart

The block diagram of the fuzzy part of the Neuro-Fuzzy system is shown in Figure 48. First, from a trained and good ANN model, the relative importance of each input over the prediction is extracted. Second, the inputs of the fuzzy c-mean are normalized using the relative importance of each input. Third, the FCM algorithm is used for clustering the dataset. Using the FCM validate indices is shown if the number of clusters are satisfactory. It is noteworthy that in this work the aim is to achieve the lowest possible number of clusters but with good performance. Therefore, the indices are analysed in order to achieve the best solution without increasing too much the number of clusters. Thus, the solution for different clusters are compared to decide the best number of clusters.

6.2.4 Training database selection from the downsized database

As said before, the main difference between hard and fuzzy clustering is that in fuzzy clustering, besides clusters, the membership to each cluster is provided. Thus, instead of using all data from each cluster, it is possible to select the experiments based on the membership.

Regarding this latter, it is proposed to design a set of rules in order to select the experiments from the clusters to train the ANN that predicts the specific grinding energy. To that end, the worst case is taken as baseline. In particular, the worst case is that the specific grinding wheel and grinding operation to predict has equal membership in all the clusters. Actually, the minimum value of membership of a cluster is the unity divided by the number of clusters.

$$N_{min} = \frac{1}{c} \tag{20}$$

where c is the number of clusters.

Once the worst case is identified, in order to select the custom training dataset to train the net, the following rules are proposed. The aim is to select specific experiments to generate the training database based on the membership:

```
IF u_ij > n x N_min THEN
   IF more than two data of the cluster have higher than n x N_min THEN
      Use these to generate training data
      IF r<e THEN
         Use data with u_ij > m x N_min
      ELSE IF u_ij > m x N_min THEN
         Use data with u_ij > m x N_min  to generate training data
         IF r<e THEN
            Use data with u_ij > N_min
         ELSE IF u_ij > m x N_min  and u_ik > N_min THEN
            Use data with u_ij > m x N_min  and u_ik > N_min to generate training data
            IF r<e THEN
               Use data with u_ij > N_min and u_ik > N_min
            ELSE IF u_ij > N_min and u_ik > N_min THEN
               Use data with u_ij > N_min  and u_ik > N_min to generate training data
               IF r<e THEN
                  Use data with u_ij > N_min  and u_ik > N_min
               ELSE IF u_ij >N_min  , u_ik > N_min  and u_il > N_min   THEN
                  Use data with u_ij > N_min, u_ik > N_min and u_il > N_min to generate training data
```

where u_{ij} is the membership of experiment x_i into cluster j, u_{ik} is the membership of experiment x_i into cluster k and u_{il} is the membership of experiment x_i into cluster l. Likewise, m and n are adjustable parameters to select wider or narrower membership ranges. Finally, r refers to the prediction error of the best network after training it and e is the worst acceptable error.

As stated before, unlike with k-mean, in c-means each data (experiments in this case) has a membership for each cluster. Thus, selecting experiments based on the membership range it is possible to select more or less experiments for the custom training dataset. Following this idea, the rules above are proposed with the aim of downsizing as much possible the custom training database, starting from a small dataset (with higher membership) to bigger datasets composed with almost all experiments under higher membership than N_{min}.

6.2.5 ANN for predicting the specific grinding energy

The methodology followed for predicting the dynamic evolution of specific grinding energy is the same described in Chapter 4. Thus, the network used is the Layer-Recurrent Neural Network with the Levenberg-Marquardt training algorithm. To improve the generalization capabilities of the net the Bayesian regularization is used and to avoid the local minima the weights and biases are updated with the Nguyen-Widrow approach.

However, in order to save training time and based on previous knowledge (see Chapter 4 and Section 6.1.3.2), instead of the two-phase methodology (see Section 4.3.3.1), the following neurons in the hidden layer (HN) and delays in the feedback (D) are used to show the efficiency of the Neuro-Fuzzy approach:

- HN ∈ {5, 7, 10}
- D ∈ {5, 7, 10}

Actually, it might think that with less training examples, the network complexity should be lower. In fact, this HN and D selection can help to analyse the relationship between the number of experiments and the complexity of the network.

Besides, it is remarkable to say that unlike in previous Section, where the aim was to generalize to new grinding wheels and grinding characteristics, in this case the aim is to estimate e_c for a specific grinding wheel and grinding characteristics. Thus, instead of using the MAME metric the maximum error of the predicted signal is used along with the MSE because there is no need of calculating the mean because only one signal is predicted with one network.

6.2.6 Discussion of the results

In order to analyse and compare the performance of the neuro-fuzzy based results, consistent indicators are necessary. Then, first of all, the results from the analysis of the fuzzy system are presented. Finally, the results for the prediction of the specific grinding energy from the downsized dataset with recurrent neural networks are discussed.

6.2.6.1 Downsizing the database using fuzzy c-means

This work has been carried out using the Fuzzy Clustering and Data Analysis Toolbox of Matlab™ (Balasko, et al., 2005). In particular, it has been used for the fuzzy c-means clustering and, also, to select the best number of clusters using the provided indices.

6.2.6.1.1 Input weighting using the relative importance of the inputs

As shown in Figure 48, first, from a good ANN model the relative importance of each input over the prediction of the specific grinding energy has to be extracted. Thus, the best network yielded in Section 6.1.3.2 is used. This network had four inputs: two characterizing the grinding wheel (grit size and the hardness) and other two describing the grinding operation (speed ratio and material removal rate).

Therefore, using the equation 18 the importance of each input over the prediction of the dynamic evolution of specific grinding energy is calculated. After, the relative importance of each input is computed (Table 18).

Input	*Relative importance (%)*
Grit size	0.42
Hardness	0.20
Speed ratio (qs)	0.04
Material removal rate (Q') (mm3/mm·s)	0.34

Table 18 The relative importance of each model input over the dynamic evolution of the specific grinding energy

Thus, it can be observed (Table 18) that the inputs with the highest relative importance are the grit size of the grinding wheel and the material removal rate. Likewise, it can be noticed that the effect of the wheel hardness is almost residual.

Once the relative importance of the inputs is computed, the inputs of the fuzzy clustering system are weighted accordingly before running the clustering. As stated above, traditionally, the inputs for fuzzy clustering are normalized within the range [0, 1]; however, in this case, the inputs are normalized using the relative importance. Since the criteria needed to deal with this normalization have not been found in the literature, the simplest way has been chosen. In fact, the normalization range of each input is easily obtained by multiplying the typical range [0 - 1] by the corresponding relative importance (Table 19):

Input	*Normalization range*
Grit size	0 - 0.42
Hardness	0 - 0.20
Speed ratio (qs)	0 - 0.04
Material removal rate (Q') (mm3/mm·s)	0 - 0.34

Table 19 FCM inputs weighted normalization ranges

6.2.6.1.2 Fuzzy clustering of the original database

In order to select the optimum number of clusters, the FCM algorithm is launched for different number of clusters. In particular, the number of clusters within the range 2 to 15 has been compared. Besides, the default fuzzy c-means algorithm parameters proposed by the toolbox developers are used i.e. weighting exponent = 2 and termination tolerance = 1e-3. Mostly, in clustering, solutions with less clusters are better. Thus, as explained in Section 6.2, using the membership it is possible to select the most suitable experiments to train custom networks.

Besides, to compare the fuzzy clustering for weighted and non-weighted inputs, the FCM algorithm is launched for both cases. In fact, first the results for the weighted inputs are shown, and second for non-weighted inputs.

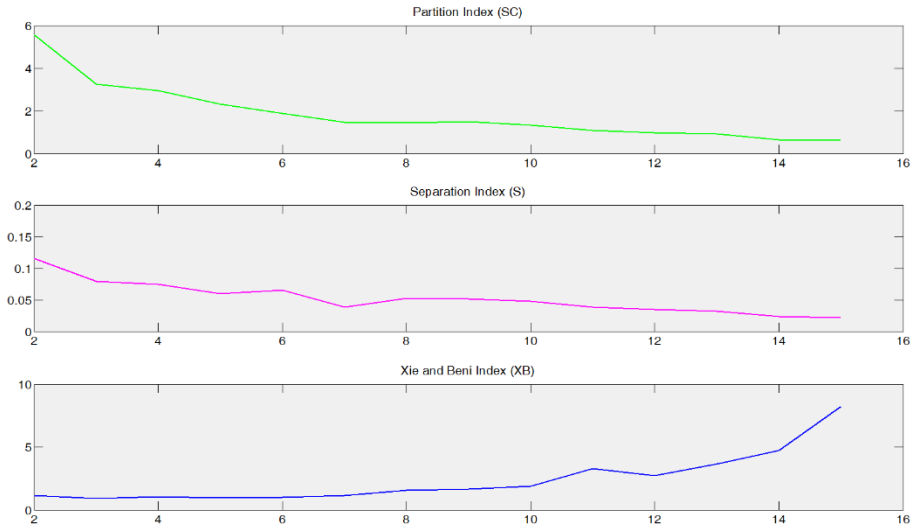


Figure 49 Values of Partition Index (SC), Separation Index (S) and Xi and Beni (XB) index for number of clusters (c) from 2 to 15 (weighted inputs)

Figure 49 and Table 20 show that for SC and S indices the results converge after *c*=7. However, after *c*=5 hardly decreases and, thus, it can be seen as the starting point of the convergence. In the case of the XB index, it reaches this local minimum at *c* = 3. However, after *c*=5, it start to increase. Considering SC, S and XB indices the optimum solutions are *c*=5 and *c*=7. However, as said before, partitions with less clusters are better, therefore, the optimal number of clusters is 5.

Clusters (c)	*SC*	*S*	*XB*
2	5.554	0.116	1.126
3	3.245	0.080	0.907
4	2.955	0.075	1.054
5	2.328	0.060	0.969
6	1.871	0.065	1.001
7	1.473	0.039	1.126
8	1.426	0.053	1.558
9	1.502	0.052	1.636
10	1.320	0.048	1.866
11	1.076	0.039	3.260
12	0.959	0.035	2.725
13	0.909	0.032	3.641
14	0.653	0.024	4.742
15	0.612	0.021	8.207

Table 20 Numerical value of Partition Index (SC), Separation Index (S) and Xi and Beni (XB) index for number of clusters (c) from 2 to 15 (weighted inputs)

In the case of the non-weighted inputs, the results are quite similar. Figure 50 and Table 21 show that after *c*=5 SC and S decrease slightly but after *c*=6 it converge. Besides,

analyzing the XB index, it reaches this local minimum at c = 6. However, from c=5 to c=7 the XB is quite similar. Therefore, in this case, although one might think that the best solution is to choose 6 cluster partitions, its results are very close to those obtained with 5 cluster partitions and thus, this latter is selected in order to have less clusters.

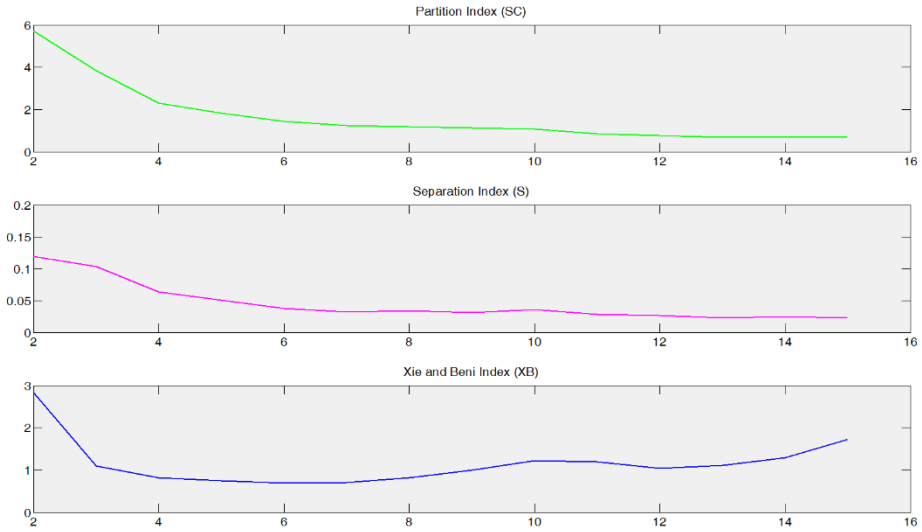


Figure 50 Values of Partition Index (SC), Separation Index (S) and Xi and Beni (XB) index for clusters from 2 to 15 (non-weighted inputs)

Clusters	*SC*	*S*	*XB*
2	5.719	0.119	2.830
3	3.839	0.103	1.102
4	2.297	0.063	0.820
5	1.815	0.051	0.750
6	1.451	0.038	0.684
7	1.238	0.032	0.706
8	1.192	0.034	0.813
9	1.141	0.031	1.001
10	1.076	0.036	1.225
11	0.864	0.028	1.187
12	0.772	0.027	1.045
13	0.681	0.023	1.103
14	0.710	0.024	1.296
15	0.691	0.023	1.721

Table 21 Numerical value of Partition Index (SC), Separation Index (S) and Xi and Beni (XB) index for clusters from 2 to 15 (non-weighted inputs)

Regarding the number of clusters, the obtained results do not yield significant differences concerning the optimum number of clusters when using or not using weighted inputs. However, respect to the results obtained with weighted clustering, the best number of clusters is 5. Meanwhile, non-weighted inputs clustering shows that with a slightly similar

number of clusters, in particular 6 clusters, the solution is similar. The reason why this happen could be that the original database is compound by experiments from a specific application field of grinding of steel parts with non-extremely demanding surface finish. Actually, only three different values are used for each input i.e. 36, 70 and 100 for the grit seize (see Section 4.2.3).

Once clusters are generated, the next step is to select the experiments to train the custom network. For the rest of the work, 5 clusters will be used for both cases i.e. weighted inputs and non-weighted inputs. Therefore, based on the equation 19 the N_{min} parameter for 5 clusters is 0.20. Actually, this is the worst case, when an experiment has the same membership for all the clusters. As described in Section 6.2.4, the next step is to select the *m* and *n* parameters of the set of rules. In this case, in view of the results yielded with 5 clusters, m=2 and n=3 are selected. In orange, the membership greater than 0.2 and lower than 0.4 is represented. Likewise, in blue the membership greater than 0.4 and lower than 0.6 is represented. Finally, in green, the membership greater than 0.6 is represented. The results show that the membership of the inputs are scattered and few of them have a membership greater than 0.8 for one cluster.

However, analysing both results (for weighted and non-weighted inputs), it can be observed that while for the weighted inputs there are more experiments with high membership (in green), for non-weighted inputs there are less experiments with high membership. Actually, the results for weighted inputs seem to be more "compact", i.e. the groups of experiments can be clearly distinguished. However, for non-weighted inputs, there are more than one experiment with more than two clusters with higher membership than 0.2 i.e. the results appear to be more scattered, less discriminatory after all. Therefore, the results confirm the conclusions obtained from the analysis of the clustering indices, which claimed that for weighted inputs 5 clusters are better than for non-weighted inputs. Actually, it seems that the groups are clearer.

Weighted

Experiment	1	2	3	4	5
1	0,11	0,31	0,13	0,39	0,07
2	0,07	0,43	0,21	0,22	0,07
3	0,07	0,30	0,38	0,14	0,11
4	0,11	0,31	0,13	0,39	0,07
5	0,08	0,52	0,09	0,25	0,06
6	0,00	0,97	0,01	0,01	0,00
7	0,06	0,49	0,26	0,10	0,09
8	0,08	0,53	0,09	0,25	0,06
9	0,00	0,98	0,01	0,01	0,00
10	0,06	0,49	0,26	0,10	0,09
11	0,08	0,52	0,09	0,25	0,06
12	0,01	0,96	0,01	0,01	0,01
13	0,06	0,49	0,26	0,10	0,09
14	0,12	0,05	0,05	0,74	0,04
15	0,14	0,10	0,32	0,31	0,13
16	0,04	0,05	0,75	0,05	0,11
17	0,12	0,05	0,05	0,75	0,04
18	0,13	0,10	0,32	0,31	0,13
19	0,04	0,05	0,76	0,05	0,11
20	0,13	0,05	0,05	0,74	0,04
21	0,14	0,10	0,32	0,31	0,13
22	0,04	0,05	0,75	0,05	0,11
23	0,79	0,02	0,04	0,10	0,05
24	0,37	0,04	0,13	0,12	0,34
25	0,09	0,04	0,17	0,06	0,65
26	0,80	0,02	0,03	0,09	0,05
27	0,37	0,04	0,13	0,12	0,34
28	0,09	0,04	0,17	0,06	0,66
29	0,79	0,02	0,04	0,10	0,05
30	0,37	0,04	0,13	0,12	0,34
31	0,75	0,03	0,04	0,11	0,07
32	0,30	0,05	0,11	0,11	0,43
33	0,04	0,02	0,08	0,03	0,83
34	0,76	0,03	0,04	0,10	0,07
35	0,30	0,05	0,11	0,10	0,43
36	0,04	0,02	0,08	0,03	0,84
37	0,75	0,03	0,04	0,11	0,07
38	0,30	0,05	0,11	0,11	0,43
39	0,04	0,02	0,08	0,03	0,83
40	0,14	0,08	0,06	0,68	0,05
41	0,13	0,16	0,29	0,28	0,15
42	0,04	0,07	0,69	0,05	0,15
43	0,14	0,08	0,06	0,68	0,05
44	0,13	0,16	0,29	0,28	0,15
45	0,04	0,07	0,70	0,05	0,14
46	0,14	0,08	0,06	0,67	0,05
47	0,13	0,16	0,29	0,28	0,15
48	0,04	0,07	0,69	0,05	0,15

Value > 0,2

Value > 0,4

Value > 0,6

Table 22 Membership range representation by colour for the 48 inputs (weighted). In orange, the membership greater than 0.2 and lower than 0.4 is represented. Likewise, in blue the membership greater than 0.4 and lower than 0.6 is represented. Finally, in green, the membership greater than 0.6 is represented

.

Non-Weighted

Experiment	1	2	3	4	5
1	0,21	0,12	0,21	0,16	0,30
2	0,15	0,12	0,16	0,15	0,43
3	0,13	0,17	0,14	0,16	0,40
4	0,20	0,12	0,28	0,17	0,22
5	0,25	0,13	0,11	0,38	0,13
6	0,17	0,14	0,08	0,50	0,12
7	0,16	0,21	0,09	0,39	0,15
8	0,22	0,11	0,10	0,49	0,08
9	0,03	0,03	0,01	0,91	0,02
10	0,12	0,21	0,08	0,49	0,10
11	0,22	0,15	0,14	0,40	0,09
12	0,15	0,17	0,10	0,51	0,08
13	0,14	0,25	0,11	0,39	0,10
14	0,21	0,10	0,24	0,09	0,35
15	0,06	0,04	0,07	0,03	0,80
16	0,10	0,14	0,12	0,08	0,56
17	0,17	0,08	0,51	0,07	0,17
18	0,10	0,09	0,41	0,05	0,35
19	0,11	0,23	0,19	0,09	0,38
20	0,16	0,11	0,53	0,09	0,12
21	0,12	0,15	0,50	0,08	0,14
22	0,13	0,27	0,27	0,12	0,20
23	0,23	0,12	0,28	0,08	0,30
24	0,15	0,12	0,19	0,06	0,48
25	0,13	0,19	0,16	0,08	0,43
26	0,18	0,09	0,51	0,06	0,16
27	0,13	0,13	0,46	0,05	0,24
28	0,13	0,25	0,23	0,08	0,30
29	0,16	0,12	0,52	0,07	0,12
30	0,13	0,16	0,50	0,07	0,14
31	0,42	0,15	0,16	0,12	0,16
32	0,32	0,24	0,13	0,12	0,20
33	0,19	0,35	0,12	0,13	0,21
34	0,49	0,14	0,18	0,10	0,10
35	0,32	0,36	0,13	0,09	0,10
36	0,13	0,57	0,10	0,09	0,11
37	0,32	0,19	0,26	0,13	0,10
38	0,23	0,34	0,21	0,12	0,10
39	0,15	0,47	0,15	0,12	0,11
40	0,45	0,12	0,12	0,16	0,14
41	0,33	0,20	0,09	0,19	0,19
42	0,17	0,32	0,10	0,19	0,22
43	0,62	0,09	0,10	0,13	0,06
44	0,36	0,32	0,07	0,18	0,07
45	0,09	0,65	0,06	0,12	0,08
46	0,32	0,17	0,22	0,20	0,09
47	0,22	0,32	0,16	0,21	0,08
48	0,13	0,47	0,12	0,18	0,10

Value > 0,2

Value > 0,4

Value > 0,6

Table 23 Membership range representation by colour for the 48 inputs (non-weighted). In orange, the membership greater than 0.2 and lower than 0.4 is represented. Likewise, in blue the membership greater than 0.4 and lower than 0.6 is represented. Finally, in green, the membership greater than 0.6 is represented

Besides, comparing the wheel characteristics and the grinding conditions of each experiment (see Table 17), as expected, experiments with the same grit size share, mostly, the same clusters in the case of weighted inputs. In fact, experiments with grit size = 100

(experiments from 23 to 39) have all of them a membership greater than 0.2 in clusters 1 and/or 5. Likewise, experiments with grit size = 36 (experiments from 1 to 13) have all of them a membership greater than 0.2 in clusters 2, 3 and/or 4. Finally, experiments with grit size = 70 (experiments from 14 to 22 and from 40 to 48) have all of them a membership greater than 0.2 in clusters 3 and/or 4. However, for the case of non-weighted inputs, the results are different. In fact, the groups are not clear. For example, the experiments 36 and 48 have a membership higher than 0.4 in the same cluster (cluster number 2), and if one looks at Table 17 the wheel characteristics and grinding conditions of both experiments, they only share the hardness and material removal rate.

6.2.6.2 Generation of custom ANNs for the prediction of specific grinding energy

As described before, 5, 7 and 10 neurons in the hidden layer (HN); and 5, 7 and 10 delays in the feedback (D) are used in order to find the best network structure for predicting the specific grinding energy. Following the strategy used in Section 6.1.2, each network structure is trained with a custom training database obtained using the FCM, initializing the weights and biases 10 times by the Nguyen-Widrow approach. Thus, now the goal of this section is to find the net that best predicts the complete dynamic evolution of specific grinding energy for specific grinding wheel and grinding operation characteristics.

In Figure 51 the lowest MSE are shown for five experiments, chosen from the original dataset, and the three types of training datasets to show the performance of the presented neuro-fuzzy approach: first, custom training dataset generated using the weighted inputs FCM results (■); second, custom training datasets generated using the non-weighted inputs FCM results (●); third, all the experiments available (▲). It is noteworthy that, of course, in the latter the only experiment not used is the one to be predicted.

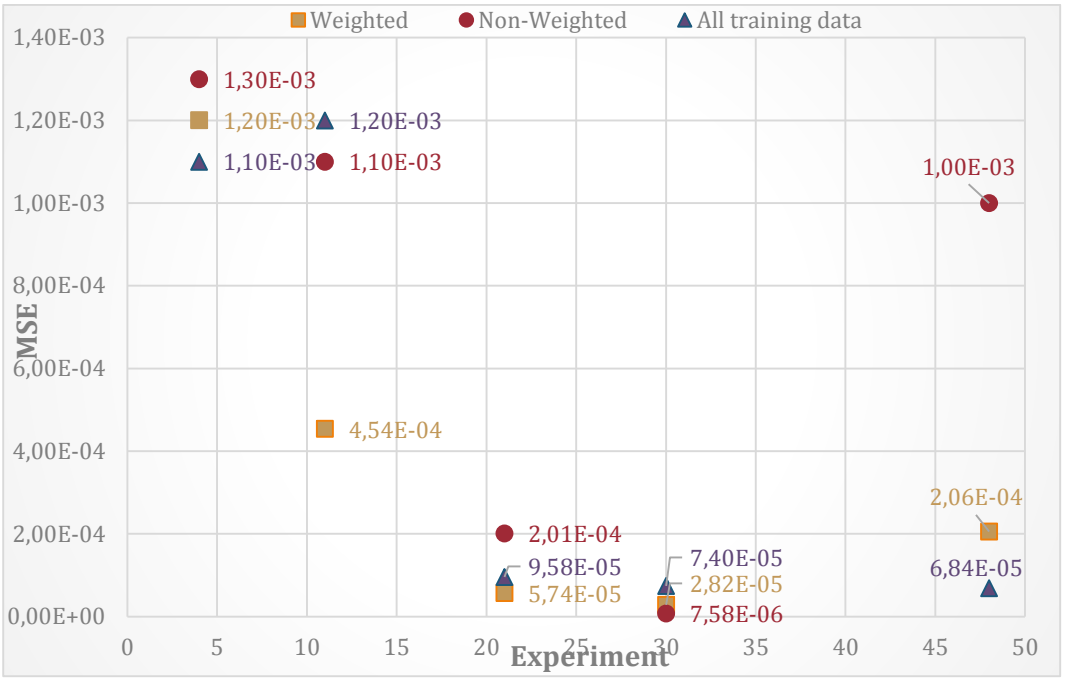


Figure 51 Summary of the lowest MSE yielded for different experiments

The results correspond to the lowest MSE in the prediction of the experiments 4, 11, 21, 30 and 48 (see Figure 51). The results clearly show that MSE values for the three training datasets are quite similar in most of the cases. However, there are some differences. For example, in the case of the prediction of the 11th experiment, the MSE results yielded with the training dataset generated with the weighted inputs are much better. Likewise, in the prediction of dynamic evolution of specific grinding energy of experiment number 48, the results yielded with the training dataset generated after FCM without weighted inputs, are far worst. Only in one case the result yielded with the training dataset generated under FCM without weighted inputs is the best one. However, although it is better, it is really close to the results yielded with all training data and weighted inputs. To sum up, in two cases the best results are achieved with the so called “weighted”, other two with all the training data available and, finally, one with the dataset generated with the non-weighted inputs.

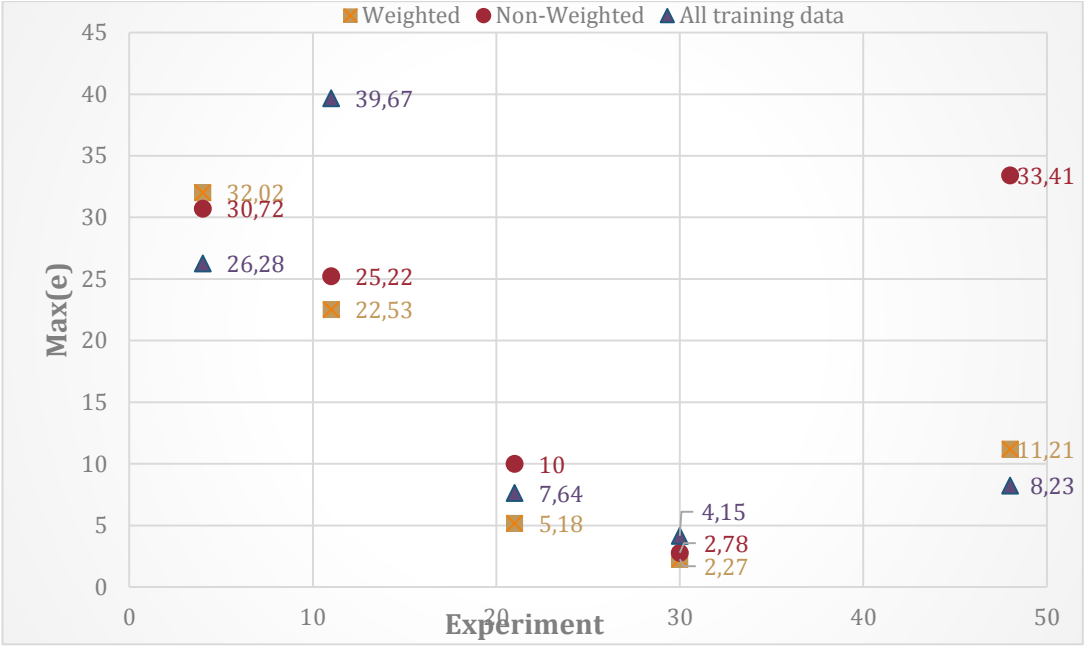


Figure 52 Summary of the lowest maximum error yielded for different experiments

However, as concluded in Section 5.3, MSE is not enough to select the best network. Therefore, in Figure 52, the lowest maximum error results are showed. It is important to recall that the network that achieves the lowest MSE is not necessarily the net that achieves the lowest maximum error. Hence, observing the results, it can be noticed that in three cases (prediction of experiments 11, 21 and 30) the best results are yielded with the custom training dataset generated with FCM and weighted inputs (■). In the remaining two cases, the lowest maximum error are for nets trained with all the training data available. Besides, the results of Figure 52 also confirm that, as said before, the MSE is not enough to select the best RNN. In fact, in the case of the prediction of the dynamic evolution of the experiment 30, the lowest MSE is yielded with the dataset generated with non-weighted inputs. Nevertheless, the lowest maximum error is yielded by a net trained with the training dataset generated with weighted inputs.

Although in some cases the best results are achieved with the training dataset composed by all training data available (except the one to be predicted), the training dataset used for training is much bigger. For example, for the prediction of the experiment 4 (see Table 24), only 4 experiments are used under the FCM weighted approach. It is noteworthy to remember that as explained in the general methodology (see Chapter 4), 10 virtual experiments are generated from the real measurements. Actually, overall, with 40 experiments a maximum error of 30.72 J/mm^3 is yielded. Using all the experiments

available (470 experiments) the maximum error reduces to 26.28 J/mm^3. In fact, the results show that it is not worth including all the experiments in the training dataset for such a small improvement.

Experiment	*Grit size*	*Hardness*	q_s	Q'
1	36	0,24	60	1
4	36	0,24	80	1
5	36	0,4	60	1
8	36	0,4	80	1
11	36	0,4	100	1

Table 24 Training dataset generated using FCM with weighted inputs and the decision proposed for the experiment 4 (in green)

Moreover, for the prediction of the experiment 11, in Table 25 the custom training dataset can be seen. Only three experiments are chosen using the FCM results with weighted inputs and the decision tree proposed. Therefore, only 30 (3x10 virtual generated) are enough to yielded the best results. Likewise, using the FCM results with non-weighted inputs and the decision tree proposed, the custom training dataset is also compounded by 30 training experiments (see Table 26). In both cases, the best network outperforms the net trained with all the experiments available (470).

Experiment	*Grit size*	*Hardness*	q_s	Q'
2	36	0,24	60	2,5
5	36	0,4	60	1
8	36	0,4	80	1
11	36	0,4	100	1

Table 25 Training dataset generated using FCM with weighted inputs and the decision proposed for the experiment 11 (in green)

Experiment	*Grit size*	*Hardness*	q_s	Q'
5	36	0,4	60	1
8	36	0,4	80	1
11	36	0,4	100	1
47	70	0,36	100	2,5

Table 26 Training dataset generated using FCM with non-weighted inputs and the decision proposed for the experiment 11 (in orange)

On the other hand, the results obtained after training the networks can lead to an analysis about the network structure (HN-D). For that purpose, in Table 27, the best networks for the prediction of different experiments and for different training datasets are showed. At

first sight, no pattern can be observed. For example, in the case of "weighted inputs", networks with 5HN, 7HN and 10HN, and 5D, 7D and 10D can be found as best. In the same way, for "non-weighted inputs", 5HN, 7HN and 10HN, and 5D, 7D and 10D can be found. Besides, not many nets can be found on the edges of the ranges (5HN5D and 10HN10D). Therefore, the preselected HN and D range seems to be the correct one. However, for the nets trained with all the data available, all the best networks have 5 delays in the feedback. Likewise, no net can be found with 10 neurons in the hidden layer. Actually, these results differ substantially from those obtained in Section 6.1.3.2, where the best two nets have 12 and 14 delays in the feedback. Therefore, the network structure yielded can lead to the conclusion that to improve the generalization capabilities of the net, as the nets of Section 6.1.3.2, more complex networks are needed i.e. with more neurons in the hidden layer (HN) and more delays in the feedback (D). Regarding this latter, it is noteworthy that the nets obtained from the neuro-fuzzy approach only generalize to one particular case, the one to be predicted.

	Weighted inputs		*Non-weighted inputs*		*All training data*	
Exp.	Best Network	Maximum error	Best Network	Maximum error	Best Network	Maximum error
4	10HU10D → Net9	32.02 J/mm^3	7HU10D → Net8	30.7 J/mm^3	7HU5D → Net5	26.28 J/mm^3
11	7HU7D → Net7	22.53 J/mm^3	10HU5D → Net2	25.22 J/mm^3	7HU5D → Net6	39.67 J/mm^3
21	7HU5D → Net9	5.18 J/mm^3	10HU7D → Net5	10.00 J/mm^3	5HU5D → Net8	7.64 J/mm^3
30	5HU5D → Net5	2.27 J/mm^3	5HU10D → Net10	2.78 J/mm^3	5HU5D → Net8	4.15 J/mm^3
48	7HU10D → Net10	11.21 J/mm^3	10HU10D → Net1	33.29 J/mm^3	7HU5D → Net6	8.23 J/mm^3

Table 27 Best networks for different experiment prediction using training dataset generated with weighted inputs (FCM), non-weighted inputs (FCM) and all data available

Finally, the MSE and maximum error results yielded are quite similar for the three different training datasets. This seems logical because, as said before, all the experiments available in the original database are from the same application field, grinding of steel parts with non-extremely demanding surface finish. However, although the results are similar, the preliminary results yielded with the weighted inputs slightly outperform the nets trained with a custom dataset obtained with non-weighted inputs. Likewise, the results achieved with all the experiments and with custom datasets are similar, better in some cases and worse in others. Therefore, it can be concluded that it is better to use

custom training datasets obtained after FCM due to the saving experimental time and money, and also, RNNs training time. However, further analysis should be carried out with an original database with experiments from different application fields in order to draw better conclusions.

6.3 Conclusions

In this Chapter a solution based on RNN to predict off-line the specific grinding energy with the ability to generalize to new wheel characteristics and grinding conditions has been presented.

The analysis of the time characteristics of the dynamic evolutions (time step and time horizon) and the number of points shows that in this kind of ANN applications, neural networks trained with dynamic evolutions with more points generalize better. However, nets trained with dynamic evolutions with lower time step (and more points for the same time horizon) do not ensure better generalization. In view of the results of the analysis of the time characteristics of the dynamic evolutions of the specific grinding energy and the number of points, 200 points are in this case enough to achieve satisfactory results. Thus, given a time horizon of 2000 mm^3/mm, the time step is set to 10 mm^3/mm.

In order to achieve the best possible ANN structure (neurons in the hidden layer and delay units in the feedback) that can predict with good accuracy the dynamic evolution of the specific grinding energy, a two-stage process which involves coarse and fine tuning is proposed. The analysis of the time characteristics of the dynamic evolutions also shows that the best results are yielded within the ranges 5 and 12 of neurons hidden layer (HN). In fact, the best results are obtained with 10 neurons in the hidden layer (HN) and 12 feedback delays (D). The MAME error for the best configuration is 16.91 J/mm^3.

The Layer-Recurrent Neural Network has shown its potential for modelling the dynamic evolution of the specific grinding energy without measuring initial real values and with only static inputs in a prediction horizon up to 2000 mm^2 of specific volume of part material removed. This is a remarkable task because it means that the proposed methodology is capable of predicting up to 200 points of a complete dynamic evolution only using static inputs without initial real values. The yielded MAME and relative errors are lower than 33.60 J/mm^3 and 23.65%, respectively, which are really acceptable errors for grinding's users. Besides, the selected RNN net generalizes with good results to new grinding conditions, 33.60 J/mm^3, and new wheels (not used during the grinding process),

15.94 J/mm^3. Thus, the net generalizes better to new wheel characteristics. In fact, it might be concluded that grinding conditions have bigger influence than wheel characteristics on the specific grinding energy prediction.

Although the proposed solution is able to predict complete dynamic evolutions without initial values for new grinding wheels and new grinding conditions, obviously, one unique RNN it is not able to predict the specific grinding energy for all the wheels commercially available. Thus, a new approach based on the generation of custom networks for specific grinding wheels and grinding conditions using available experiments is proposed.

In particular, a neuro-fuzzy proposal is presented to generate custom networks for specific grinding wheels and grinding conditions downsizing the experiments database. Besides, in grinding, carrying out grinding experiments is a highly time and resource consuming task. Therefore, cutting down on these experiments and selecting the custom training dataset is highly recommended and a desirable step forward in order to generate custom models for grinding process variables. Among different clustering techniques such as k-means, in this work fuzzy c-means are used. Unlike k-means, in fuzzy clustering, each datum can belong to more than one cluster with a membership level and it is possible to select the experiments based on the membership. Therefore, for a better selection of the custom training dataset based on membership a set of rules is proposed. Finally, unlike other approaches for reducing the dataset using fuzzy c-means, in this work the inputs are weighted in order to "help" the FCM by extracting knowledge from trained ANN weights and, thus, "illuminating" the black box.

The FCM results show that for the FCM with weighted inputs the optimum clusters are 5 while without weighted inputs the optimum clusters are 6, which is very close to 5. Therefore, for the following analysis 5 clusters are used. From the analysis of the memberships it can be concluded that the results for weighted inputs appear to be more "compact" than for non-weighted inputs. Moreover, comparing wheel characteristics and grinding conditions of each experiment, as expected, experiments with the same grit size share, mostly, the same clusters in the case of weighted inputs. In fact, these results clearly show that weighting the inputs with previous knowledge can guide the clustering.

Under custom training datasets (weighted and non-weighted) and non-custom (all the experiments available), the results of training in five experiments show that, although the MSE and maximum errors for the three training datasets are quite similar in most of the cases, the custom networks obtained with the weighted approach yields slightly better

results. Likewise, the results achieved with all the experiments and with custom datasets are similar, better in some cases and worse in others. This seems logical because, as said before, all the experiments available in the original database are from the same application field, grinding of steel parts with non-extremely demanding surface finish. Therefore, it can be concluded that it is better to use custom training dataset obtained after FCM due to the saving experimental time and money, and also, RNNs training time. However, further analysis should be carried out with an original database with experiments from different application fields in order to draw better conclusions.

7 MODELLING DYNAMIC EVOLUTIONS WITH SPIKING NEURAL NETWORKS

Although the results obtained in Section 5 and Section 6 are satisfactory in terms of accuracy from the grinding process perspective, it can be noticed that the objective of predicting with high accuracy up to 200 points is really ambitious. Actually, there is still much room for improvement. At the sight of the results, one could think that the classical artificial neural networks are not powerful enough for modelling the complex relationship between the grinding wheel characteristics and operating conditions, and the grinding variables such as wheel wear, surface roughness or specific grinding energy. Actually, studying different issues found in the literature for the selected architecture, one could think in two major problems: one, the vanish gradient problem due to the number of delays and use of the BPTT, and, two, the underfitting, due to lack of training samples and complexity of the grinding process. Therefore, in order to overcome this problem the so-called third generation neural networks, the Spiking Neural Networks (SNNs), are proposed because they are more suited for modelling dynamic evolutions, due to the temporal encoding of the spikes. However, the use of SNNs for modelling dynamic evolutions is not easy yet because there is no a proper method to encode analog data into spikes and reconstruct the original data precisely. Likewise, for time series forecasting, in each time step the SNN must fire at least one spike in order to have an analog predicted value in the output of the net. Thus, in this work a new encoding/decoding algorithm based on the pulse-width modulation is proposed in order to use SNNs for modelling dynamic evolutions. The results show that the proposed algorithm can encode and decode analog signals precisely. Moreover, with the proposed algorithm the spiking neuron is capable to fire one spike in each time, something fundamental fore time series forecasting.

7.1 Introduction

The results obtained with the LRNN applying the general methodology described in Chapter 4, and under the initial perspective of using commercial software and well known and well-established ANN architectures and training algorithms, are promising and better than those yielded with analytic models. Moreover, the adopted perspective has allowed to identify and to understand the implications of the problem, keeping these separated from the complexity of new architectures and/or training algorithms. However, although the results are good enough for developing soft sensors or providing information about the performance of a specific wheel in advance, these models are far from being used for control. In fact, as supposed, grinding is a very complicated process to model due to the semi-handmade production of the grinding wheels.

Although the results provided by the smart sensors are satisfactory in terms of accuracy from the grinding process perspective, it can be noticed that the objective of predicting with high accuracy up to 200 points is really ambitious and, actually, after one point the so-called constancy phenomenon arises. Obviously, this is a not desirable phenomenon because a sensor should provide an accurate measure during the whole machining operation. Regarding the results provided by the ANN model for the estimation of the dynamic evolution of specific grinding energy, the results are also satisfactory from the grinding process perspective. However, there is still much room for improvement.

At the sight of the results, one could think that the classical artificial neural networks are not powerful enough for modelling the complex relationship between the grinding wheel characteristics and operating conditions, and the grinding variables such as wheel wear, surface roughness or specific grinding energy. Moreover, the goal of predicting dynamic evolutions without any initial real value is really challenging.

Actually, studying different issues found in the literature for the selected architecture, one could think in two major problems: one, the vanish gradient problem due to the number of delays and use of the BPTT (Hochreiter & Schmidhuber, 1997), and, two, the underfitting, due to lack of training samples and complexity of the grinding process (Bishop, 2006). As stated in Section 4.2.3, performing grinding experiments is a highly cost and time consuming task, thus, increasing significantly the training dataset does not seem the best solution. Therefore, it seems more reasonable to find more powerful neural networks to model the grinding process with higher accuracy. Among different network

structures two stand out over others for modelling temporal data. The first one is Long short-term memory (LSTM) recurrent neural network (Hochreiter & Schmidhuber, 1997) used in Deep Learning with excellent results in fields like Natural Language Processing (NLP) or video recognition. The biggest advantage of this network is that it avoids the vanish gradient. The other networks are the so-called third generation neural networks, the Spiking Neural Networks (SNNs). These networks try to mimic human neurons in a more realistic way. Besides, in these networks the data is encoded in spikes. Actually, due to the temporal encoding of the spikes, the SNNs have inherently the capacity to manage temporal data (Grüning & Bohte, 2014). This characteristic make the SNNs more suited for modelling dynamic evolutions like the dynamic evolution of the wheel wear, the surface roughness and the specific grinding energy. Therefore, in order to obtain better models of surface roughness, wheel wear and specific grinding energy, the SNNs modelling of dynamic evolutions is proposed in the following Chapter.

7.2 Spiking Neurons

The basic idea behind the Artificial Neural Networks of the second generation such as the perceptron neuron and the spiking neurons (third generation) is basically the one described in Chapter 2: computing systems whose central theme is borrowed from the analogy of biological neural networks (Mehrotra, et al., 1997).

However, real biological neurons communicate with each other using electrical pulses called “spikes” (Grüning & Bohte, 2014). The general process of spiking signal transmission is illustrated in Figure 54 (Grüning & Bohte, 2014): First, action potentials travel along axons and activate synapses. Then, these synapses release a neurotransmitter that quickly diffuses to the post-synaptic neuron. In the post-synaptic neuron, these neurotransmitters affect the membrane potential of the neuron. Excitatory Postsynaptic Potentials (EPSPs) increase the membrane potential, and without new inputs, this excitation then leaks away with a typical time constant. On the other hand, Inhibitory Postsynaptic Potentials (IPSPs) decrease the membrane potential. When sufficient EPSPs arrive at a neuron, the membrane potential may reach a certain threshold, and the neuron generates a spike itself, resetting its membrane potential. Thus, the generated spike then travels on to other neurons.

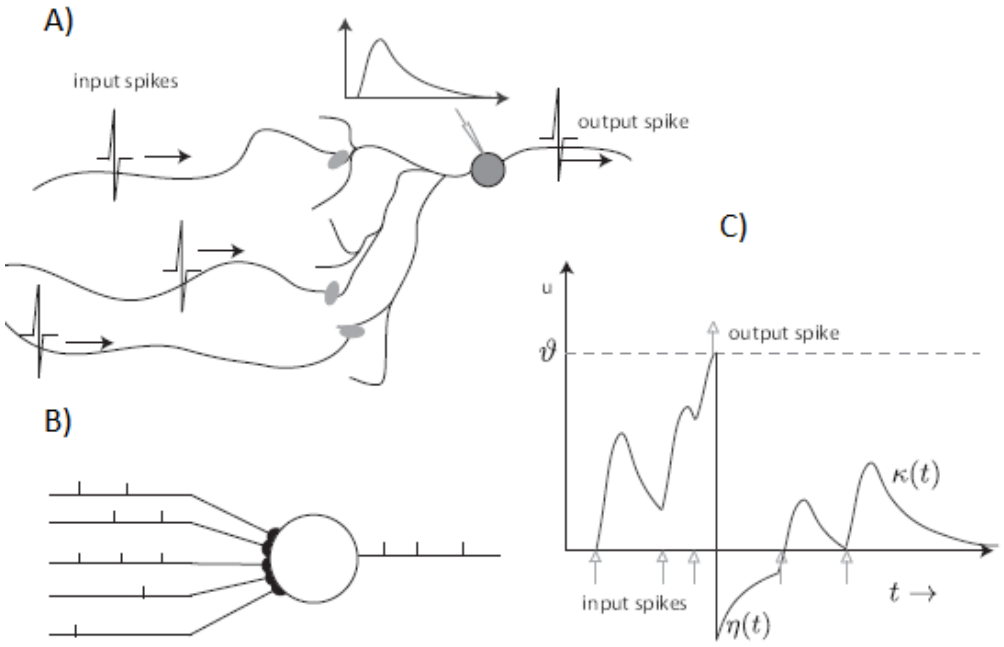


Figure 53 A) Spikes arrive from other neurons at the synapses of the postsynaptic neuron. Its membrane potential rises quickly with each incoming spike, and then slowly decays again (inset). However, if several spikes arrive in a short time window, the membrane potential may reach a certain threshold, and a spike is fired down the axon. B) Schematically, incoming spikes on various dendrites produce timed spikes responses as the output. C) Schematic response of the membrane potential to several spikes arrive. If the threshold θ is crossed, the membrane potential is reset to a low value, and a spike fired (Grüning & Bohte, 2014)

The spikes have an amplitude of about 100 mV and typically a duration of 1-2 ms (see Figure 54). A chain of spikes emitted by a single neuron is called a spike train, a sequence of stereotyped events which occur at regular or irregular intervals (Gerstner & Kistler, 2002). Since all spikes of a given neuron look similar, the form of the spikes does not carry any information. Moreover, it is the number and the timing of spikes which matter. The action potential or spike is the elementary unit of signal transmission (Gerstner & Kistler, 2002).

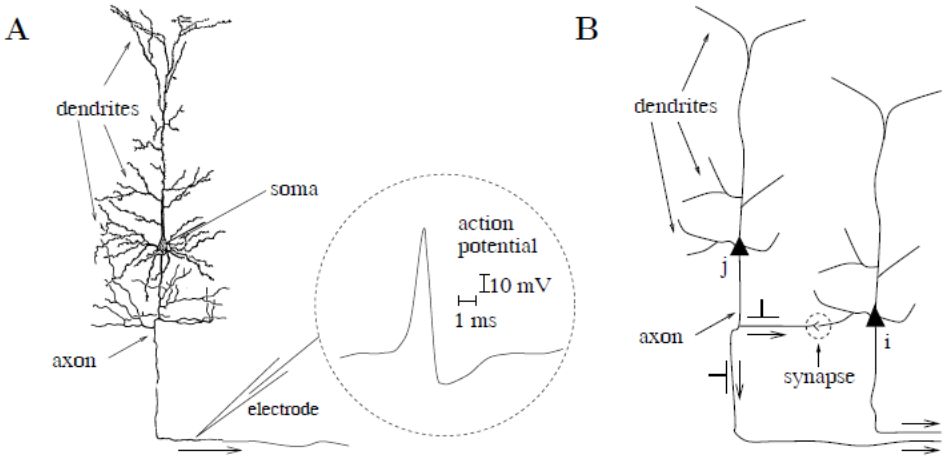


Figure 54 A) Single neuron in a drawing by Ramon y Cajal. Dendrite, cell body or soma, and axon can be clearly seen. The inset shows an example of a neuronal action potential, it is a short voltage pulse of 1-2 ms duration and an amplitude of about 100 mV. B) Signal transmission from a presynaptic neuron *j* to a postsynaptic neuron *i*. (Gerstner & Kistler, 2002)

The above is a simplified explanation of spiking neurons behaviour. In real life, neurons show many different spike behaviours. To explain these behaviours, detailed models have been developed. These models are typically expressed as dynamical systems of various complexity, and include models like the Leaky-Integrate-and-Fire (LIF) model, the Izhikevich model (Izhikevich, 2003) or the Hodgkin-Huxley model (Hodgkin & A. F. Huxley, 1952).

7.3 Signal encoding into spike trains

As explained above, in spiking neural networks the "information" is transmitted as spikes. Actually, this requires that meaning is given to spike trains to transmit it to spiking neurons. The first dominant paradigm about neural representation of the neural information was the idea that it is encoded in the firing rate (Ponulak & Kasinski, 2011). The firing rate code has been the dominant paradigm in neurophysiology and, also, in artificial neural networks. However, recent neurophysiological research suggests that, in some neural systems the efficient neural information processing is more probable to be based on the precise timing of action potentials (Ponulak & Kasinski, 2011). A special pattern of temporal code is polychronous group where spike-trains with fixed mutual timings are distributed across a group of neurons (Izhikevich, 2006).

In real-world, almost all signals are analog in nature. Therefore, when spiking neural networks are used to process analog signals, these must be converted into spikes. This has been much less researched by neuroscience because it was not needed for doing neuron analysis (Schrauwen & Van Campenhout, 2003). Some encoding methods algorithms proposed for precise timing paradigm are as follows:

- Hough Spiker Algorithm (HSA): The basic idea behind this algorithm is to try to do a reverse convolution of the stimulus (Hough, et al., 1999). Actually, the idea is that if the impulse response of the linear filter is smaller or equal than the input, then there has to be a spike in order to produce this (Schrauwen & Van Campenhout, 2003). It is possible to reconstruct the original signal using this algorithm (see Figure 55).

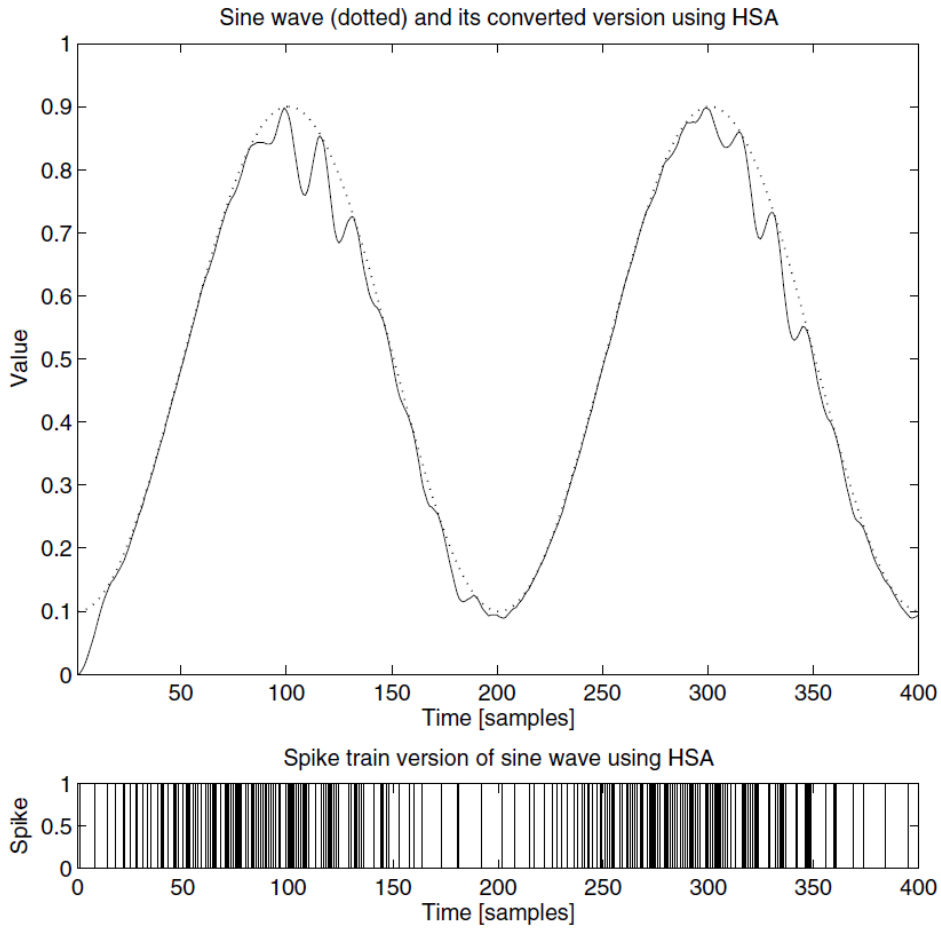


Figure 55 Sine wave encoded into spikes and reconstructed using HSA. In the upper plot the original sine wave can be seen in dotted line, and the converted version in solid line. The lower plot visualizes the spike train (Schrauwen & Van Campenhout, 2003)

- Bens Spiker Algorithm (BSA): Like the HSA algorithm, this algorithm also assumes the use of a FIR reconstruction filter. Every instant in time τ, the algorithm calculates two error metrics. If the first error is smaller than the second minus a threshold, then fire and subtract the filter from the input, else do nothing (see Figure 56) (Schrauwen & Van Campenhout, 2003).

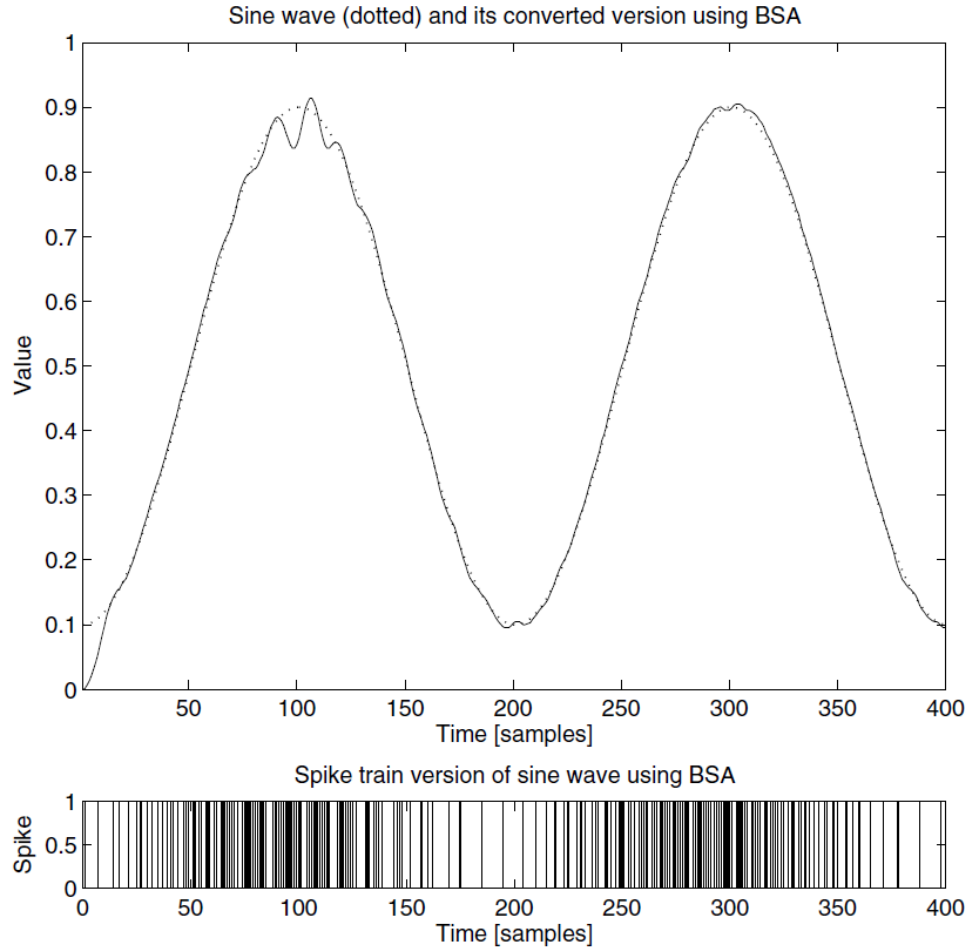


Figure 56 Sine wave encoded into spikes and reconstructed using BSA. In the upper plot the original sine wave can be seen in dotted line, and the converted version in solid line. The lower plot visualizes the spike train (Schrauwen & Van Campenhout, 2003)

- Step-Forward Spike Encoding Algorithm (SF): For a given signal $S(t)$ where (t = 1, 2. . . n), a baseline $B(t)$ variation during time t is defined with $B(1) = S(1)$. If the incoming signal intensity $S(t_1)$ exceeds the baseline $B(t_1-1)$ plus a threshold defined as Th, then a positive spike is encoded at time t1, and $B(t_1)$ is updated as $B(t_1) = B(t_1-1) + Th$; and if $S(t_1) <= B(t_1-1) - Th$, a negative spike is generated

and $B(t_I)$ is assigned as $B(t_I) = B(t_I-1)-Th$. In other situations, no spike is generated and $B(t_I) = B(t_I-1)$ (Kasabov, et al., 2016).

7.4 Spiking Neural Networks learning

As in artificial neural networks, one of the main characteristics of the spiking neural networks is their capacity to learn from examples. In biological neural networks, it is thought that the basic mechanism behind learning and memory capacity of the neurons is the ability of the synaptic weights to change their strength (Baudry, 1998). Various models of learning for spiking neural networks have been proposed for unsupervised and supervised learning:

7.4.1 Unsupervised training algorithm

In spiking neural networks the typical example of unsupervised learning algorithm is the Hebbian learning based Spike-timing-dependent plasticity (STDP). In STDP if the postsynaptic neurons fires shortly after the presynaptic neuron has fired the synaptic is strengthen, and if the presynaptic neuron fires shortly before postsynaptic neurons fires, the synaptic is weakened. For further information see (Song, et al., 2000). The above form of STDP is just one out of many physiological forms of STDP.

7.4.2 Supervised training algorithm

Supervised training algorithms for precise time coding paradigm are still highly uncovered by existing approaches (Ponulak & Kasinski, 2011). Moreover, to date, there is no general-purpose supervised training algorithm for spiking neural networks (Grüning & Bohte, 2014). One of the first algorithms presented was the Spike Prop (Bohtea, et al., 2002). The method is based on a gradient descent equivalent to the traditional backpropagation algorithm (see Section 3.1.5.2). In SpikeProp, the error is calculated as the temporal difference between the actual spike and the desired or target spike. The major limitation of the SpikeProp is that it can deal only with one spike firing during a single simulation cycle. However, some years later other two algorithms based on SpikeProp were proposed that could deal with multiple spikes (Booji & Nguyen, 2005) (Ghosh-Dastidar & Adeli, 2009). Again, later, other supervised training algorithms were proposed that could lead with multiple spikes. However, these algorithms can only work with one spike neuron connected to multiple spike trains. In particular, the ReSuMe algorithm was biologically plausible and the results show that was efficient learning spike sequences

and classifying temporal spike patterns (Ponulak & Kasiínski, 2010). Finally, another supervised learning algorithm for SNN called SPAN that enables a single neuron to learn spike pattern associations was proposed (Mohemmed, et al., 2012).

7.5 Modelling dynamic evolutions with Spiking Neural Networks

Due to the temporal encoding of the spikes, the SNNs have inherently the capacity to manage temporal data and, thus, they are more suited for modelling dynamic evolutions. SNNs have been widely used for classifying temporal-data using unsupervised or supervised learning. Thus, for example, for ultra-fast image recognition a feedwordward network with STDP learning algorithm was used (Thorpe, et al., 2001). Likewise, other unsupervised models have been proposed for image recognition (Shin, et al., 2010), image compression and reconstruction (Perrinet & Samuelides, 2002), detection and classification of visual objects (Guyonneau, et al., 2004) and odor recognition (Finelli, et al., 2008). Besides, spiking neurons have been also used in supervised data classification of spike patterns (Nikolic, et al., 2009), epilepsy detection (Ghosh-Dastidar & Adeli, 2009) or speech recognition (Gütig & Sompolinsky, 2009). Finally, more recently, a morphologic framework based on spiking neurons have been presented for modelling spatio-temporal data such as seismic data, early stroke or video recognition (Kasabov, et al., 2016).

However, much less effort has been made in order to model dynamic evolutions such as one-step ahead, multi-step ahead forecasting or complete dynamic evolutions. Only few works of the same research team can be found (Reid, et al., 2014) (Reid, et al., 2015). All the works are based on polychronization (Izhikevich, 2006). In fact, these works perform a classification task rather than a true forecasting task. Moreover, there is not analog data reconstruction and the output resolution is dependant on the number of spike neurons.

So in conclusion, as the literature review shows, SNNs are suitable for modelling time dependant data like dynamic evolutions. However, few works can be found that solve this task. Actually, the main reason is that so far there is no a proper method to encode analog data into spikes and reconstruct the original data precisely. Likewise, for time series forecasting, in each time step the SNN must fire at least one spike in order to have an analog predicted value in the output of the net. Thus, in this work a new encoding/decoding algorithm is proposed in order to use SNNs for modelling dynamic evolutions.

7.5.1 Pulse-width modulation based spike encoding

The proposed technique for encoding analog signal into spikes and reconstruct the original signal is based on the well-known Pulse-width Modulation (PWM). Therefore, first the PWM technique is explained. Then, the proposed technique is presented. Finally, the results are shown.

7.5.1.1 Pulse-width Modulation

Pulse-width modulation (PWM) is one of the most commonly used ways to perform analog-to-digital conversion in applications of diverse areas, including motor control signal processing, communication and power electronics (Meng, et al., 2016).

The conventional PWM uses sinusoidal modulating signal types as reference signal and generally a sawtooth or triangular signal, as a carrier signal. The carrier wave frequency must be sufficiently large compared to that of the reference signal. This technique is represented in Figure 57. The basic idea is that both signals, reference and carrier, are compared in order to obtain a modulated signal. As is represented in Figure 57, if the reference signal amplitude is higher than the carrier signal amplitude, the modulated signal is represented with a high amplitude rectangular pulse. In the contrary, if the reference signal amplitude is lower than the carrier signal amplitude, the modulated signal is represented with low amplitude. Therefore, the modulated signal is a quadratic signal with different pulse-widths. In the same way, it should be noted that by comparing the modulated signal with the carrier signal it is easy to reconstruct again the reference signal.

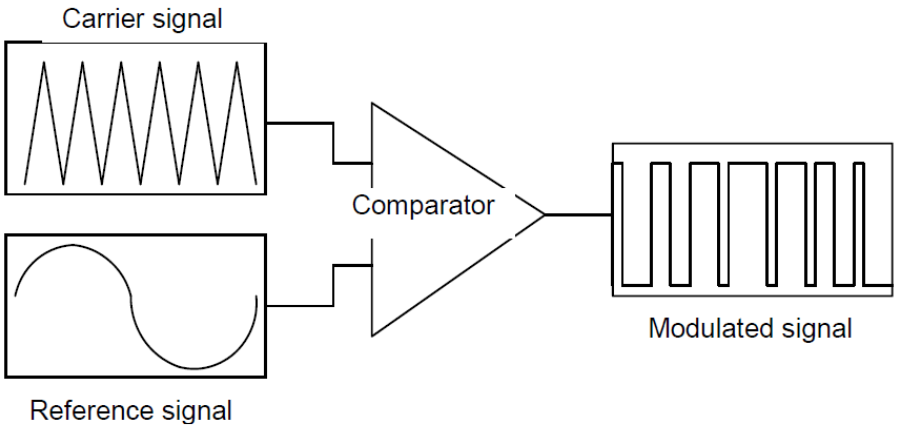


Figure 57 Diagram of conventional PWM. A carrier signal and the reference signal are compared in order to generate a modulated signal (Arab, et al., 2014)

Therefore, knowing the basic idea behind the Pulse-width Modulation method, in the following Section the modification of this method to obtain a spike train and reconstruct the original analog data is presented.

7.5.1.2 Pulse-width Modulation for Spiking Neurons

Based on the idea behind the PWM of using a carrier signal and a comparator, an encoding method to convert analog signals into spikes is presented. Actually, the basic idea behind the encoding method presented here is to translate the slope of the analog signal into the time of the spike train. In this case, the so-called reference signal of the PWM is substituted by the original analog signal (see Figure 58 A). Then, following the PWM methodology the original analog signal and the carrier are compared (see Figure 58 B). Actually, if the carrier signal is higher than the original signal the comparison result is 1. In the contrary, if the carrier signal is lower than the original signal the comparison result is 0 (see Equation 21). Therefore, the result of the comparison is a quadratic signal with different widths (see Figure 58 C). Finally, to generate the spike train, each rising edge of the quadratic signal represents one spike (see Figure 58 D).

$$s(t) \geq c(t) \qquad p(t) = 0 \tag{21}$$

$$s(t) < c(t) \qquad p(t) = 1$$

Where s is original analog signal, c is the carrier signal, p is the quadratic signal and t represents the time.

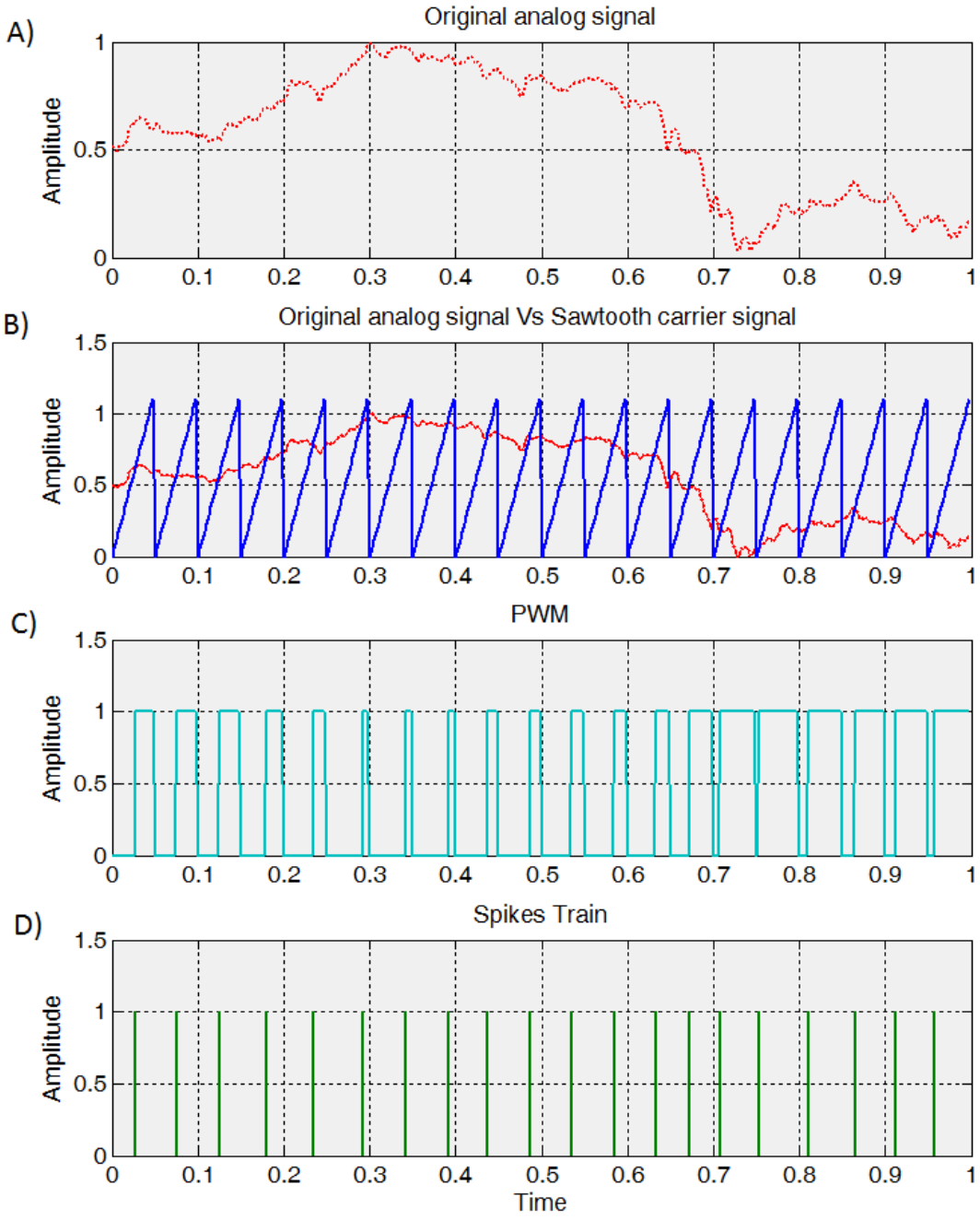


Figure 58 A) Original analog signal normalized within the range [0, 1]. B) A comparison between the original analog signal and the sawtooth carrier signal. C) The result of the comparison is a quadratic signal with different width. D) Each rising edge of the quadratic signal is one spike

As can be noticed, the basic idea behind the encoding algorithm is quite simple: it is basically a comparison between a carrier and the original signal. Likewise, to reconstruct the original signal the process is quite similar. First, the spike train is compared with the same carrier signal used for encoding (see Figure 59 A and B). The intersection between

the carrier signal and the corresponding spike is the point of the reconstructed original signal (see Figure 59 C). Then, generally speaking, the original signal can be reconstructed by interpolating.

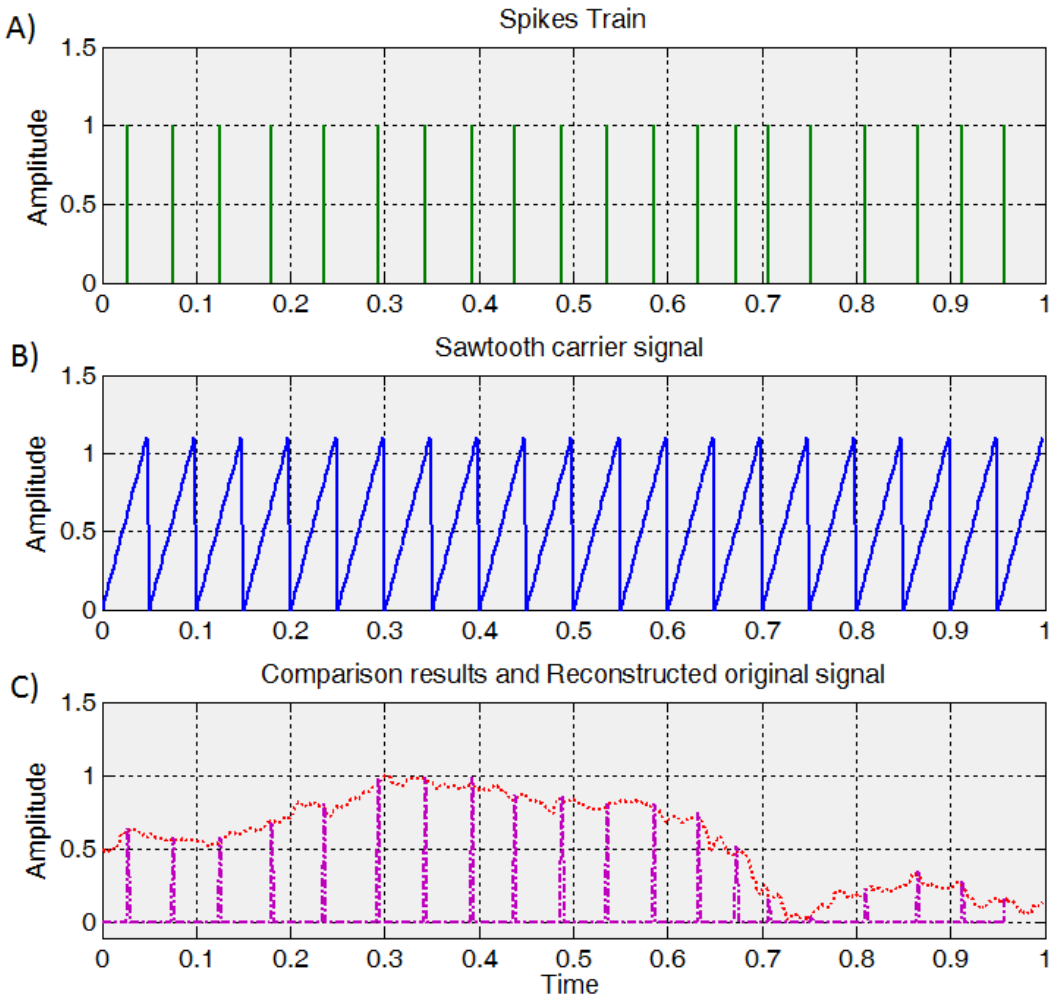


Figure 59 A) Spike train. B) Same carrier signal used for encoding. C) The point of the carrier signal that a spike crosses (purple) is a point of the reconstructed original signal (red)

Above the basic idea of the encoding method is described, however, there are two important parameters that have a great influence over the reconstruction accuracy:

a) the number of carrier waves (nc), which is directly related to the carrier pulse width.
b) the number of points per carrier wave (npc).

The number of carrier waves (nc) makes reference to the number of sawtooth in the carrier signal. For example, the nc of the sawtooth carrier signal in Figure 59 is 20. In order to make decision about the value of this parameter, it can be taken into consideration the sampling rate (indeed, the sampling requirements) of the original data. As it is well-known, the Nyquist–Shannon sampling theorem was proposed by Nyquist and demonstrated by Shannon (Shannon, 1949). The theorem states that a bandlimited baseband *x(t)* within bandwidth *B* can be exactly reconstructed from its sample values by low-pass filtering if the sampling rate is higher than 2*B*. Later, in (Huang, et al., 2011) they show that by passing the PWM waveform with frequency 2*B* and voltage ±1 through an ideal low-pass filter with cutoff *B* frequency, it is possible to get back the exact original bandlimited signal *x(t)* within bandwidth *B*. Actually, in classical sampling, sampling is a process of multiplying the analog signal *x(t)*, with a sampling signal *s(t)*, which is a train of impulses (Dirac delta) (see Figure 60). Indeed, one value is captured per impulse. PWM, on the other hand, represents a signal by using pulses of constant amplitude but variable widths and, in that sense, PWM is a substitute for classical sampling (Huang, et al., 2011). Actually, making an analogy with classical sampling, in which "one value is captured per impulse", in the PWM based encoding algorithm "one spike is generated per carrier pulse". In other words, and conceptually speaking, the sampling signal in the PWM based encoding algorithm is the carrier instead of the train of impulses (see Figure 61). Taking into account that a proper selection of the sampling rate is essential to guarantee a proper signal reconstruction, the number of carrier waves nc (i.e. the width of the carrier pulse) should be directly related to the sampling requirements of the signal to be encoded. In particular, given an already acquired signal, and assuming that it has been acquired at a properly selected sampling rate, the hypothesis is that nc equal to the number of points acquired (minus one) should provide a satisfactory recovery accuracy.

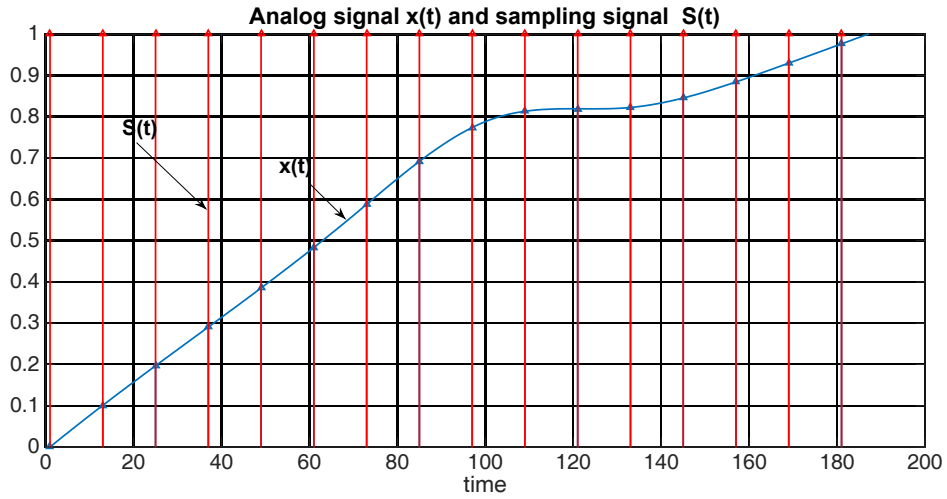


Figure 60 Classical sampling

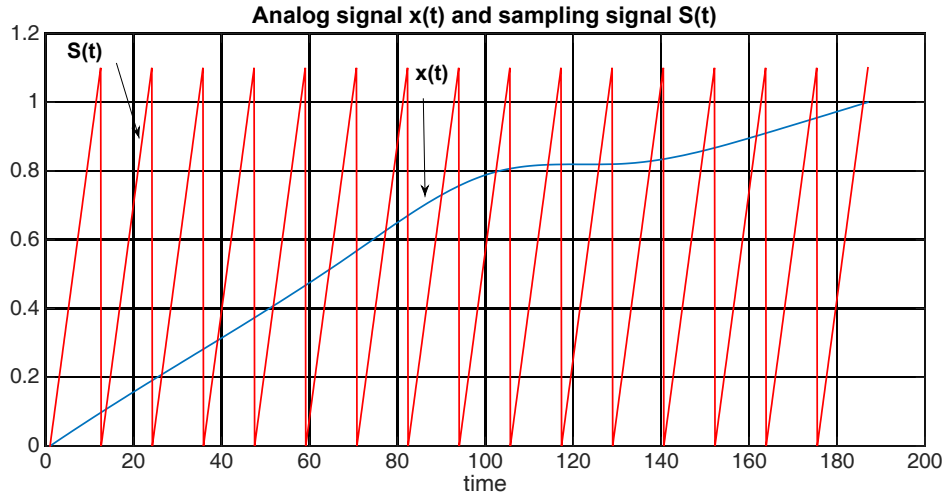


Figure 61 PWM based encoding "sampling"

This means that with the algorithm presented here it is possible to get back the exact original by properly adjusting the number of carrier waves, i.e. the sampling rate.

Likewise, it should be highlighted that for the particular case of forecasting applications it is necessary to have one value at each time step. Actually, with the encoding methods of the state of the art presented in Section 7.3, it was not possible to perform true forecasting with SNN because those methods cannot generate one spike per time step. However, using the encoding method proposed here it is possible to generate one spike each time step by setting the nc equal to the number of time steps of the time series.

On the other hand, the number of points per carrier wave npc makes reference to the number of points per carrier pulse used to generate one spike. In this case, it can be taken

into consideration another important parameter in classical sampling: the resolution. As it is well-known, the resolution is the smallest detectable change in the signal and has a direct impact on the recovery accuracy. Making again an analogy with classical sampling, in which the more resolution the better reconstruction since smaller changes in the value of the signal are captured, in the case of the PWM based encoding algorithm, the smallest detectable change is determined by the number of points within the carrier pulse. To illustrate this idea, in Figure 62 and Figure 63 the reconstruction can be seen for different number of points (npc) with the same number of carrier waves (nc). In fact, the hypothesis in this case is that the more npc, the better accuracy. The results show that when the number of points per carrier is higher the accuracy of the reconstruction increases. Of course, this means that, depending on the application requirements, the lowest possible resolution should be selected so as not to increase excessively the number of points of the carrier.

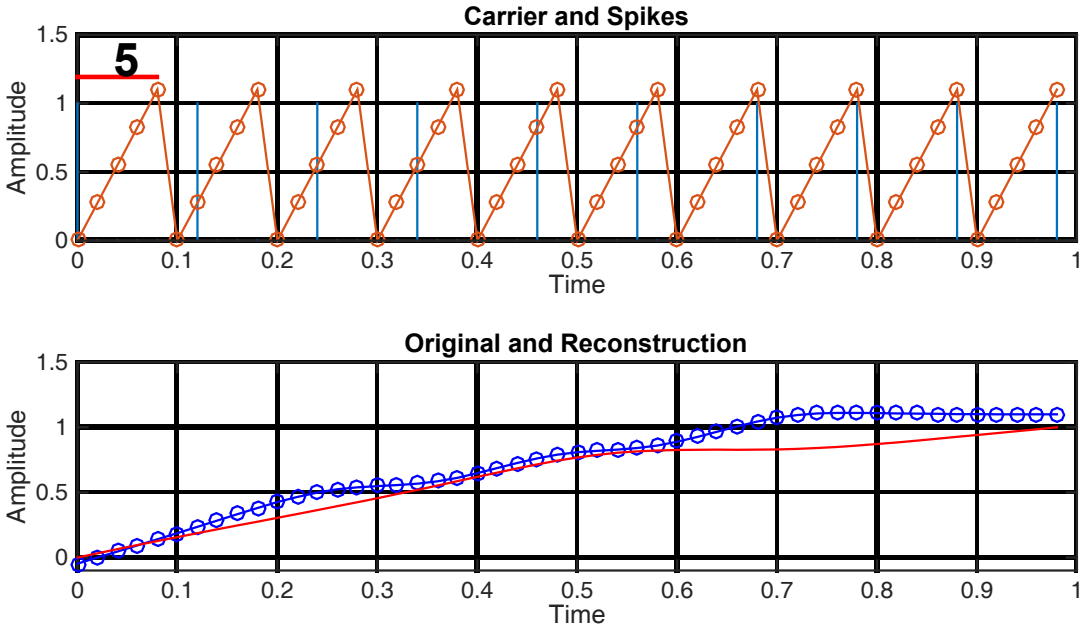


Figure 62 Original analog signal ('-' red) reconstruction ('-o' blue) with npc=5

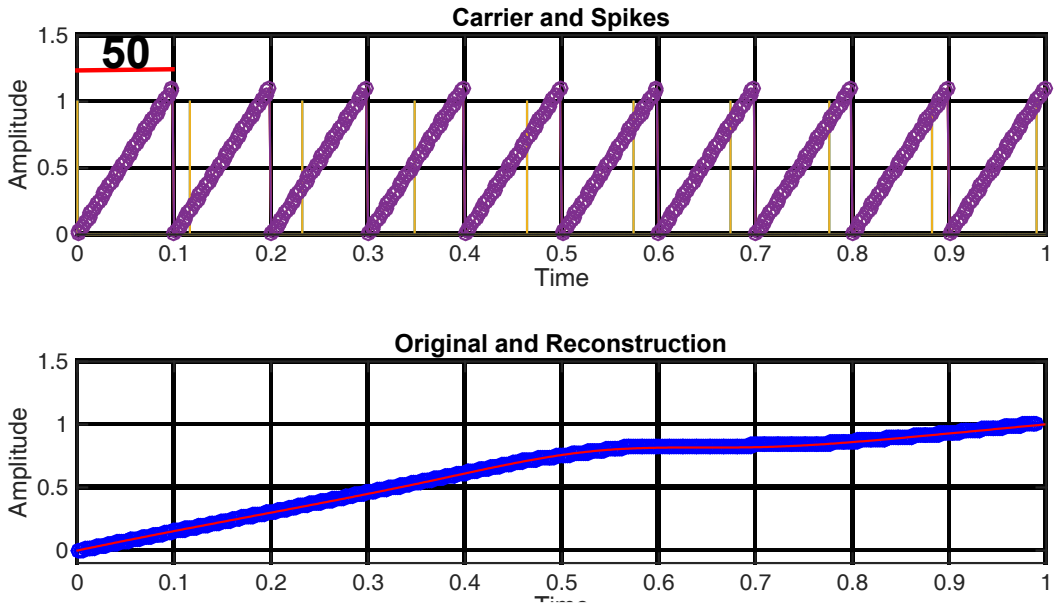


Figure 63 Original analog signal ('-' red) reconstruction ('-o' blue) with npc=50

7.6 Results

In order to demonstrate the effectiveness of the proposed encoding methodology, the encoding and reconstruction of different analog signals with different number of carrier waves (nc) and number of points per carrier wave (npc) are shown in this Section. Hence, the encoding and reconstruction of one analog signal of wheel wear, surface roughness and specific grinding energy are shown. In particular, the wheel wear, surface roughness and specific grinding energy of 82AA36G6VW grinding wheel with operation characteristics of material removal rate (Q') ($mm^3/mm \cdot s$) equal to 2.5 and speed ratio (q_s) equal to 60.

In Figure 64 the MSE results yielded for wheel wear are presented. The results clearly show that with more points per carrier wave the results are better i.e. with a higher resolution the results are better. However, adding more carrier waves does not mean an improvement in the reconstruction of the signal. In fact, the better results are achieved with npc=80 and nc=40. That means that after one point, increasing the sample rate (number of carrier waves) does not improve the encoding and the reconstruction of the original signal.

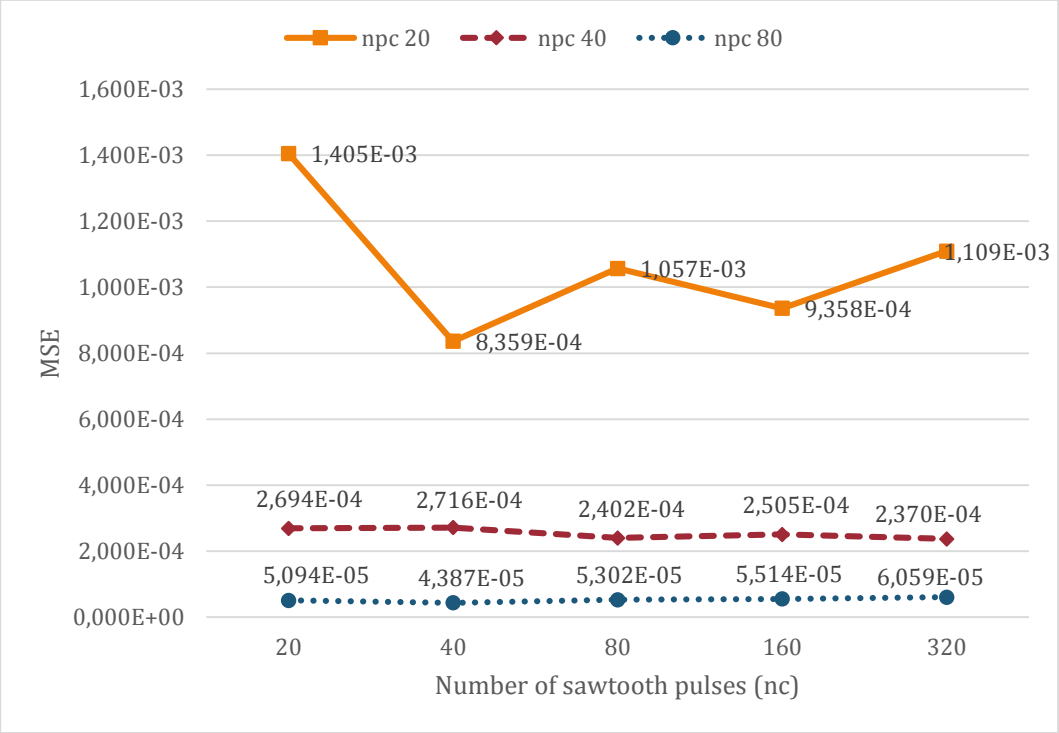


Figure 64 MSE results yielded for wheel wear analog signal of 82AA36G6VW q_s=60 Q'=2.5 for different combinations of nc and npc

On the other hand, the MSE error between npc=40 and npc=80 are minimum. Moreover, Figure 65 and Figure 66 show that the reconstruction in both cases is excellent, being the difference between them minimum. Therefore, depending of the application and the analog signal selecting lower nc and nps, although the results are a bit worse, is better because less number of points are added to the original signal. In these two cases, the number of points of the encoded signal with nc=40 and npc=80 is 3200 whilst for the encoded signal with nc=20 and npc=40, the number of points is 800, four times lower.

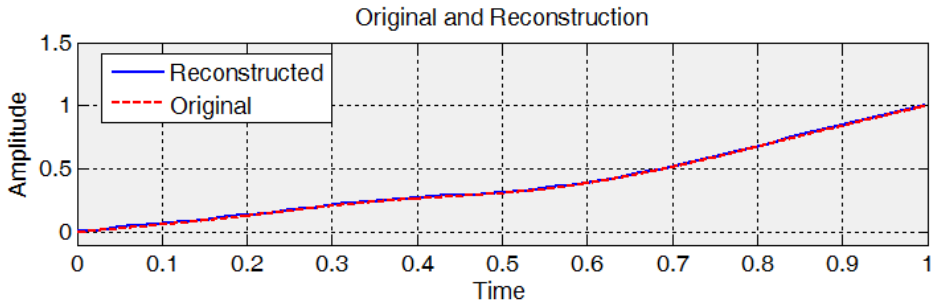


Figure 65 Wheel wear original and reconstructed signal with nc=40 and npc=80

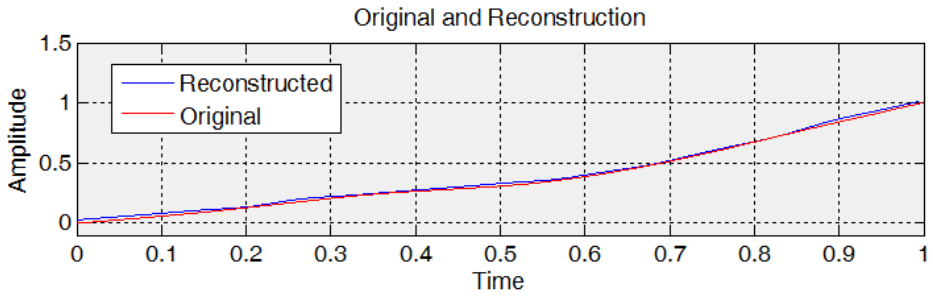


Figure 66 Wheel wear original and reconstructed signal with nc=20 and npc=40

The MSE results yielded for surface roughness are quite similar (see Figure 67). As for the wheel wear, the results clearly show that with more points per carrier wave the results are better. However, adding more number of carrier waves does not mean an improvement in the reconstruction of the signal. In fact, the better results for npc=40 and npc=80 are yielded with nc=40 and nc=20, respectively. Besides, the second lowest MSE error for npc=20 is achieved with nc=20. That means that after nc=40, for npc=40 and npc=80 the encoding and the reconstruction of the original signal does not improve.

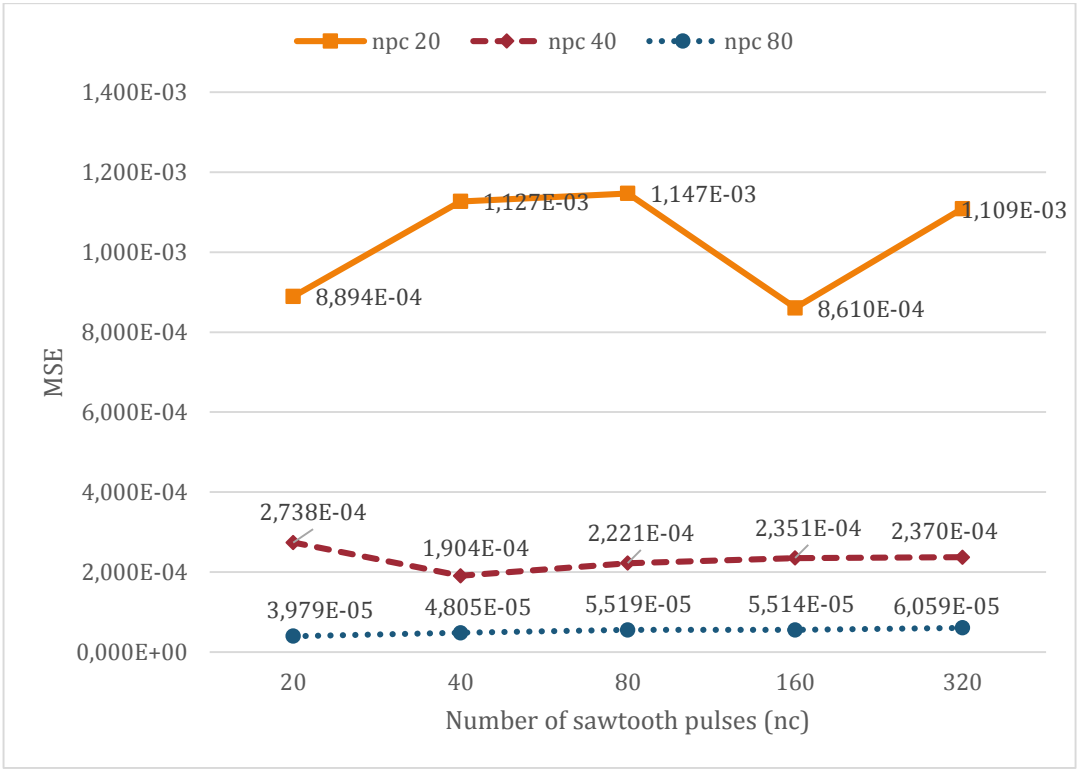


Figure 67 MSE results yielded for surface roughness analog signal of 82AA36G6VW q_s=60 Q'=2.5 for different combinations of nc and npc

In Figure 68 and Figure 69 similar comparison done for wheel wear is shown. In this case, also, the difference between the best solution (nc=20 and npc=80) and the solution with npc=40 and minimum number of points in the spike train is minimum. In both cases the reconstruction is accurate. However, the number of points of the encoded signal with nc=20 and npc=80 is 1600 whilst for the encoded signal with nc=20 and npc=40, the number of points is 800, two times lower.

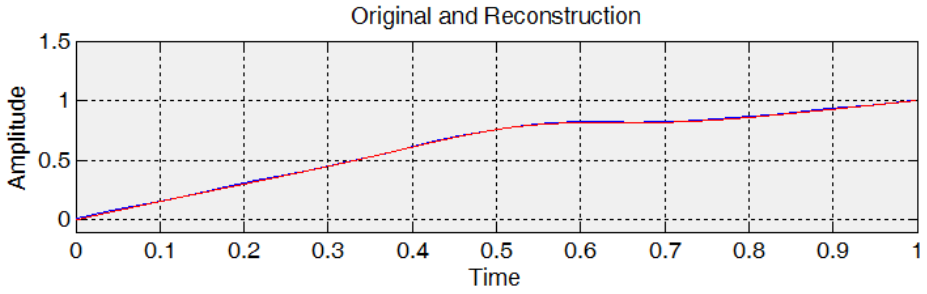


Figure 68 Surface roughness original and reconstructed signal with nc=20 and npc=80

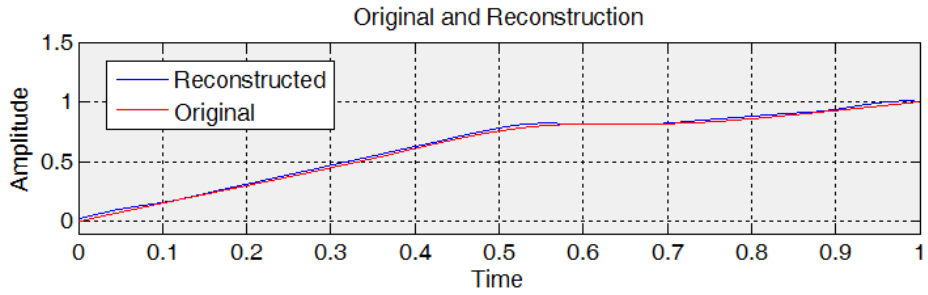


Figure 69 Surface roughness original and reconstructed signal with nc=20 and npc=40

However, the MSE results yielded for encoding and reconstructing the specific grinding energy results are a bit different (see Figure 70). The specific grinding energy signal has higher frequency components than wheel wear and surface roughness signals. Therefore, the sampling rate (nc) needed to reconstruct the original signal is higher. For example, the best result is yielded with nc=160 and npc=80 whilst the worst results for npc=20 and npc=80 are, clearly, with nc=20. That confirms that signals with higher frequency components need more carrier waves for better reconstruction.

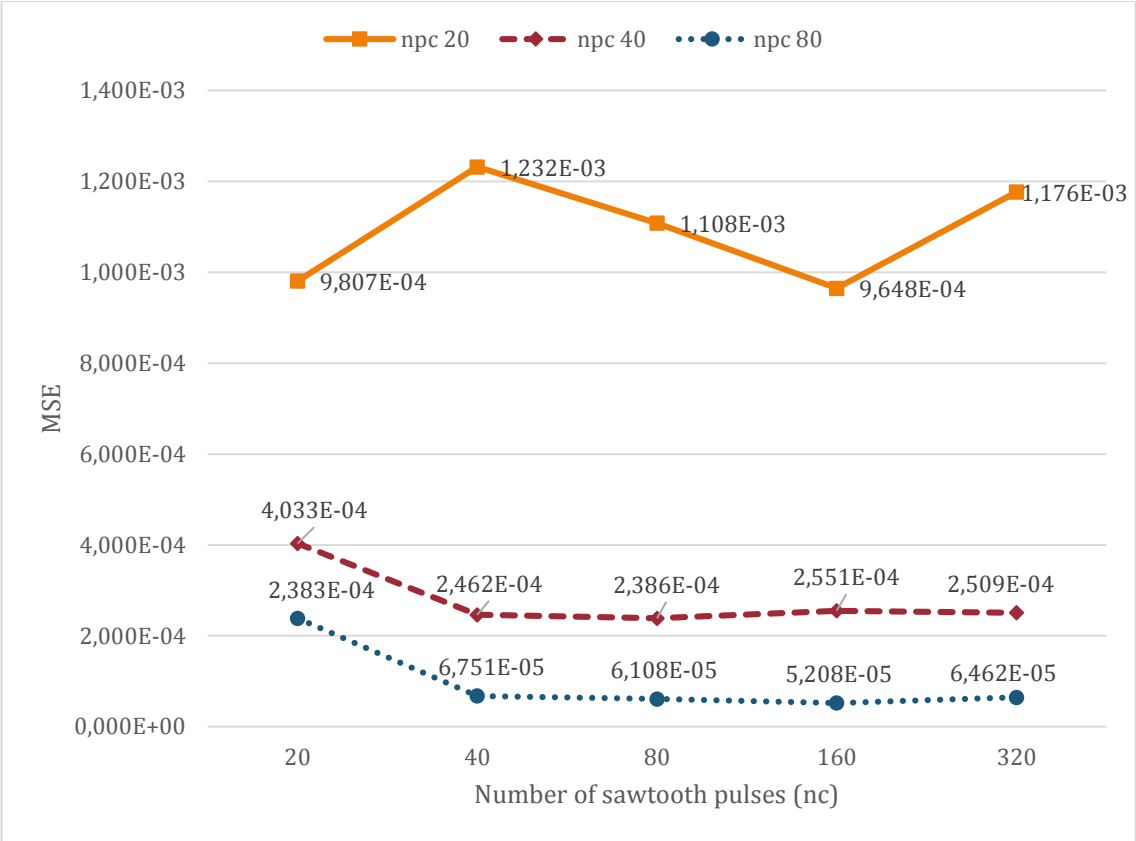


Figure 70 MSE results yielded for specific grinding energy analog signal of 82AA36G6VW q_s=60 Q'=2.5 for different combinations of nc and npc

However, although the MSE difference between the best result (nc=160 and npc=80) and the worst results with npc=40 (nc=20) seems to be big, the reconstruction result of both signals is good (see Figure 71 and Figure 72). The reconstruction with nc=160 and npc=80 is almost perfect. Likewise, while the reconstruction with nc=20 and npc=40 is not perfect, it is quite good. Besides, in this case the number of points of the spike train between both cases is higher, 12800 against 800, sixteen times higher. Therefore, depending of the application field, required accuracy and/or number of training samples, in some cases it is worth using lower nc and npc.

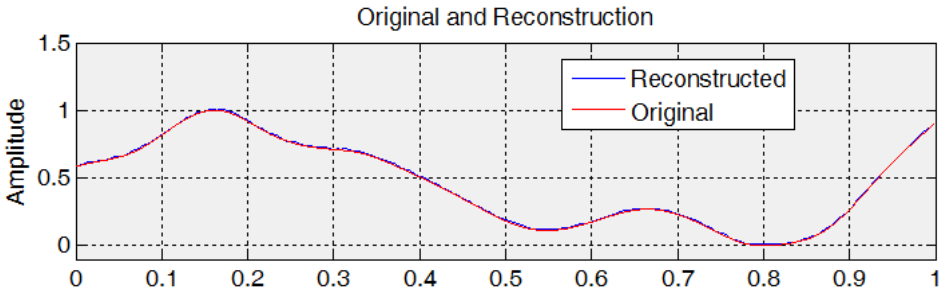


Figure 71 Specific grinding energy original and reconstructed signal with nc=160 and npc=80

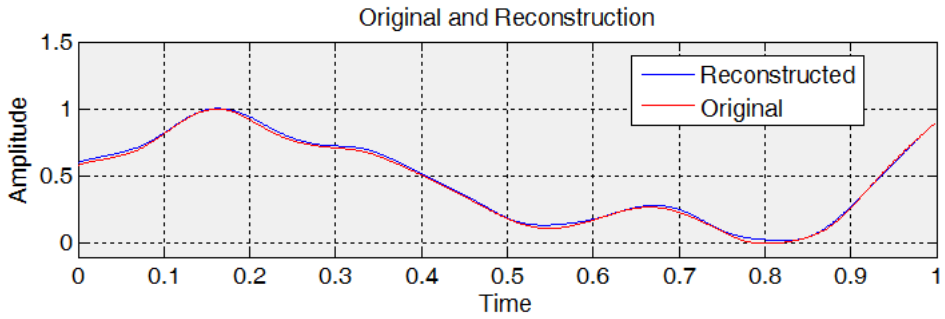


Figure 72 Specific grinding energy original and reconstructed signal with nc=20 and npc=40

Finally, it is important to appoint that the results with npc=20 are by far the worst results for the three signals (wheel wear, surface roughness and specific grinding energy). Moreover, the results show a huge difference between different numbers of carrier signal pulses (nc), and in any case converge to a solution. Therefore, npc= 20 is not recommended for encoding these analog signals.

7.7 Conclusions

Although one of the main objectives of the presented work was the use of using commercial software and well known and well-established ANN architectures and training algorithms, and the results obtained with the LRNN applying the general methodology described in Chapter 4 are promising and better than those yielded with analytic models, at the sight of the results, one could think that the classical artificial neural networks are not powerful enough for modelling the complex relationship between the grinding wheel characteristics and operating conditions, and the grinding variables such as wheel wear, surface roughness or specific grinding energy.

Actually, studying different issues found in the literature for the selected architecture, one could think in two major problems: one, the vanish gradient problem due to the number of delays and use of the BPTT, and, two, the underfitting, due to the lack of training samples and complexity of the grinding process. Thus, the next step is to find more powerful neural networks to model the grinding process with higher accuracy.

Among different network structures, the so-called third generation neural networks, the Spiking Neural Networks (SNNs), stand over others due to the inherent capacity to manage temporal data and, thus, they are more suited for modelling dynamic evolutions, because of the temporal encoding of the spikes.

Despite of SNN being widely used for classifying temporal-data using unsupervised or supervised learning, much less effort has been made in order to model dynamic evolutions such as one-step ahead, multi-step ahead forecasting or complete dynamic evolutions. Only few works that perform a classification task rather than a true forecasting task can be found. The main reason is that so far there is no a proper method to encode analog data into spikes and reconstruct the original data precisely. Likewise, for time series forecasting, in each time step the SNN must fire at least one spike in order to have an analog predicted value in the output of the net. Thus, in this work a new encoding/decoding algorithm is proposed in order to use SNNs for modelling dynamic evolutions.

The proposed technique for encoding analog signal into spikes and reconstruct the original signal is based on the idea behind the PWM of using a carrier signal and a comparator. Actually, the basic idea behind the encoding method presented here is to translate the slope of the analog signal into the time of the spike train. The results for wheel wear and surface roughness clearly show that with more points per carrier wave the results are better i.e. with a higher resolution the results are better. However, adding more number of carrier waves does not mean an improvement in the reconstruction of the signal. On the other hand, the MSE results yielded for encoding and reconstructing the specific grinding energy results are a bit different. The specific grinding energy signal has higher frequency components than wheel wear and surface roughness signals. Therefore, the sampling rate (nc) needed to reconstruct the original signal is higher.

On the other hand, the reconstruction of the original in the three signals with less number of carrier waves (nc=20) and less number of points per carrier wave (npc=40) is quite

good. Besides, with lower nc and npc values the number of points in the spike train decreases. Therefore, depending of the application field, required accuracy and/or number of training samples, in some cases it is worth using lower nc and npc. Finally, results with npc=20 are by far the worst results for the three signals (wheel wear, surface roughness and specific grinding energy) and not recommended for encoding these analog signals.

8 CONCLUSIONS

Machining tool processes are nowadays key technologies in a competitive global manufacturing marketplace. Due to its capacity for producing parts of high precision and high surface quality in difficult-to-machine materials, grinding is widely used in high-added value sectors. Thus, grinding process modelling is of primary importance in order to improve the accuracy and save time investment.

For modelling the grinding process, two main research lines can be highlighted, the theoretical models approach and the approach based on intelligent systems. However, industrial application of the theoretical models is not an easy task yet due to the lack of precise information about the composition and performance of the wheel and the relations between the different process variables and process outputs are highly non-linear. Therefore, many researchers use intelligent techniques in order to model the grinding process.

However, industrial application of intelligent techniques in grinding has been, also, very limited so far because almost all the research works that use intelligent techniques found in the scientific literature provide particular solutions for a given wheel-workpiece pair. In few cases more than one grinding wheel or workpiece are used. Results cannot be generalized, in no case, to other types of grinding wheels not used during the design of the models. Besides, in the cases of prediction of surface roughness, the output of the model is the value related to the current state of wear of the grinding wheel being this value considered unique for the whole grinding process.

Likewise, all the works that model the wheel wear use signals like Acoustic Emissions (AE), vibration signals or audio signals. However, accessibility of the machining area is limited due to the very aggressive conditions and low accessibility. Besides, in most of the cases, the given solution is binary, the dull or sharp condition of the wheel. Thus, the solutions do not give the actual status of the grinding wheel.

Finally, little effort has been dedicated to the modelling of the specific grinding energy with intelligent techniques. However, it is a fundamental variable in order to know the performance of the grinding process and it is also useful for estimating the power requirement of the grinding machine.

Therefore, in order to model the actual status (related to the current state of wear of the grinding wheel) of the surface roughness, the wheel wear and the specific grinding energy it is necessary to take those as dynamic evolutions i.e. systems that have a dynamic behaviour and actual events depend on past events. When dealing with this type of systems, ANNs have shown that are suitable and widely used for modelling those dynamic evolutions.

The review of the state-of-the-art shows that in most of the cases the prediction of dynamic evolutions is based on the past or historical events. Thus, the ANN is trained with past historical data for predicting future events. For the one-step prediction, feedforward neural networks or recurrent neural networks are used. In fact, although the RNN are more powerful, in one-step forecasting the feedforward neural networks have shown great performance. However, for multi-step ahead the problem is more complex. Thus, for multi-step ahead the recurrent neural networks are more powerful because it is said that the RNNs have "memory" thanks to the feedbacks.

Aside from modelling dynamic evolutions using historical past values, in other cases, the objective is to predict the future events using for training initial values. The training strategy is almost the same. However, some application fields require a third approach that is totally different. In this approach, historical or initial data are not available. In those cases, the aim is to predict a complete dynamic evolution but without initial or real values. Thus, the training strategy is completely different. First, the inputs are different from the output of the network. Second, the prediction cannot describe neither one-step ahead neither multi-step ahead because a new complete dynamic evolution is predicted. Another important aspect to highlight is the generalization: given the nature of the undertaken applications, the scope of generalization in the literature review is limited to the target dynamic evolution, i.e. usually only one dynamic evolution is used for training the net and another one is predicted.

Therefore, in order to model the specific grinding energy, wheel wear and surface roughness, it is crucial to predict a complete dynamic evolution without measuring any initial real value. Besides, it is crucial to break with the wheel-workpiece pair in order to develop more industrial useful models. Thus, under the selected application field, because it is reasonable to think that one ANN will not be capable to model all the grinding wheel and grinding conditions, a methodology for generating new ANNs for different

application fields should be developed with generalization capabilities concerning new grinding wheels and new grinding conditions.

The proposed methodology is based on the knowledge about the grinding process extracted from analytical models widely recognized. Actually, these analytical models relate the specific grinding energy, the wheel wear and the surface roughness between them and with the wheel characteristics and cutting conditions. As a result of the analysis of the analytical models, the inputs of the ANN model are selected. These have to represent the wheel characteristics and the grinding conditions. Therefore, based on the application field selected, grit size and wheel hardness are chosen to characterize the grinding wheel while specific material removal rate (Q') and speed ratio (q_s) are used to characterize the grinding process.

Concerning the selection of the ANN architecture, the feedforward ANN architecture is discarded because it is not possible to model the dynamic evolution with static inputs with feedforward neural networks. Among different RNNs the Elman based LRNN is selected because it functions without any initial output values. Besides, the well-known Levenberg-Marquardt training algorithm is selected for its fast convergence. Likewise, to improve the generalization, the Bayesian regularization is selected for its good generalization capabilities and because in Bayesian regularization training and testing datasets are enough. Finally, the hyperbolic tangent activation is chosen as activation function of the hidden neurons and Nguyen-Widrow approach for initialization of the network weights and biases to improve the learning speed.

The next key point is the training and testing database configuration. First, the data is pre-processed in order to obtain periodic dynamic evolutions. Then, the training and testing datasets are selected. From the 46 experiments available, 4 are selected for testing the generalization capabilities of the trained network. Finally, due to the lack of samples to train the network, virtual experiments are generated. In this case, 10 experiments are generated from each of the 42 training experiments.

The last part of the presented methodology is how to select the best network. Thus, in this work a two-phase methodology is proposed. First, the phase called “coarse tuning” is done. During this phase the aim is to obtain the network structure (HN-D) around which the lowest MSE test errors are yielded. Given the best structures inferred in the coarse tuning, the so called “fine tuning” is carried out in HN-D structure taken one by one within the specified range. During the fine tuning, the MAME metric is used.

Based on the proposed RNN methodology, soft sensors to measure the wheel wear and surface roughness during the process with the ability to generalize to new wheel characteristics and grinding conditions are developed. The calibration process of the sensor involves establishing the best possible ANN structure (neurons in the hidden layer and delay units in the feedback) that can model with good accuracy the dynamic evolution of wheel wear or surface roughness. This is carried out in a two-stage process proposed in the methodology.

However, it is not possible to select the best ANN structure only by comparing MSE and MAME values of the test dataset. Thus, a new *ad-hoc* indicator is proposed: the CV of the final stage of the prediction horizon. However, the CV analysis is highly dependent on the application and modelling signal and, consequently, the reference values to discern if a signal remains constant or not depends on those characteristics.

The selected recurrent neural architecture, *Layer-Recurrent Neural Network*, has shown the potential for modelling the dynamic evolution of the wheel wear and surface roughness without measuring initial real values in a prediction horizon up to 2000 mm^2 of specific volume of part material removed. This is a remarkable task because it means that the proposed methodology is capable of predicting up to 200 points of a complete dynamic evolution without initial real values. Besides, the network configurations that optimally represent the wheel wear and surface roughness confirms that for modelling the dynamic evolutions of both wheel wear and surface roughness, the quantity of hidden neurons and delay units is quite similar. Therefore, this confirms the initial hypothesis accepted by literature and expressed in Equations (1) and (2).

Regarding the estimation of the evolution of wheel wear the highest maximum error for wheels used and not used during the training process are 9 μm and 67 μm, respectively. Regarding the estimation of the surface finish, the selected net generalizes with good results to new wheels (not used during the grinding process), 0.36μm maximum error, and new grinding conditions, 0.32μm maximum error.

After developing soft sensors to estimate on-line the wheel wear and surface roughness, a solution based on RNN to predict off-line the specific grinding energy with the ability to generalize to new wheel characteristics and grinding conditions is carried out.

The analysis of the time characteristics of the dynamic evolutions (time step and time horizon) and the number of points shows that in this kind of ANN applications, neural networks trained with dynamic evolutions with more points generalize better. However,

nets trained with dynamic evolutions with lower time step (and more points for the same time horizon) do not ensure better generalization. In view of the, given a time horizon of 2000 mm^3/mm, the time step is set to 10 mm^3/mm. In order to achieve the best possible ANN a two-stage process is followed. The yielded MAME and relative errors are lower than 33.60 J/mm^3 and 23.65%, respectively, which are really acceptable errors for grinding's users. Besides, the selected RNN net generalizes with good results to new grinding conditions, 33.60 J/mm^3, and new wheels (not used during the grinding process), 15.94 J/mm^3.

Although the proposed solution is able to predict complete dynamic evolutions without initial values for new grinding wheels and new grinding conditions, obviously, one unique RNN it is not able to predict the specific grinding energy for all the wheels commercially available. Thus, a neuro-fuzzy proposal is presented to generate custom networks for specific grinding wheels and grinding conditions downsizing the experiments database. Besides, in grinding, carrying out grinding experiments is a highly time and resource consuming task. Therefore, cutting down on these experiments and selecting the custom training dataset is highly recommended and a desirable step forward in order to generate custom models for grinding process variables.

Among different clustering techniques such as k-means, in this work fuzzy c-means are used. Unlike k-means, in fuzzy clustering, each datum can belong to more than one cluster with a membership level and it is possible to select the experiments based on the membership. Therefore, for a better selection of the custom training dataset based on membership a set of rules is proposed. Finally, unlike other approaches for reducing the dataset using fuzzy c-means, in this work the inputs are weighted in order to "help" the FCM by extracting knowledge from trained ANN weights and, thus, "illuminating" the black box.

Under custom training datasets (weighted and non-weighted) and non-custom (all the experiments available), the results of training in five experiments show that, although the MSE and maximum errors for the three training datasets are quite similar in most of the cases, the custom networks obtained with weighted approach yields slightly better results. Likewise, the results achieved with all the experiments and with custom datasets are similar, better in some cases and worse in others. This seems logical because, as said before, all the experiments available in the original database are from the same application field, grinding of steel parts with non-extremely demanding surface finish. Therefore, it can be concluded that it is better to use custom training dataset obtained after FCM due

to the saving experimental time and money, and also, RNNs training time. However, further analysis should be carried out with an original database with experiments from different application fields in order to draw better conclusions.

Although one of the main objectives of the presented work was the use of commercial software and well known and well-established ANN architectures and training algorithms, and the results obtained with the LRNN applying the general methodology are promising and better than those yielded with analytic models; at the sight of the results, one could think that the classical artificial neural networks are not powerful enough for modelling the complex relationship between the grinding wheel characteristics and operating conditions, and the grinding variables such as wheel wear, surface roughness or specific grinding energy.

Actually, one could think in two major problems: one, the vanish gradient problem due to the number of delays and use of the BPTT, and, two, the underfitting, due to the lack of training samples and complexity of the grinding process. Thus, the next step is to find more powerful neural networks to model the grinding process with higher accuracy. Among different network structures the so-called third generation neural networks, the Spiking Neural Networks (SNNs) stand over others due to the inherent capacity to manage temporal data because of the temporal encoding of the spikes.

Despite of SNN being widely used for classifying temporal-data using unsupervised or supervised learning, much less effort has been made in order to model dynamic evolutions. The main reason is that so far there is no a proper method to encode analog data into spikes and reconstruct the original data precisely. Likewise, for time series forecasting, in each time step the SNN must fire at least one spike in order to have an analog predicted value in the output of the net. Thus, in this work a new encoding/decoding algorithm is proposed in order to use SNNs for modelling dynamic evolutions.

The proposed technique for encoding analog signal into spikes and reconstruct the original signal is based on the idea behind the PWM of using a carrier signal and a comparator. The results for wheel wear and surface roughness clearly show that with more points per carrier wave the results are better i.e. with a higher resolution the results are better. However, adding more carrier waves does not mean an improvement in the reconstruction of the signal. On the other hand, the MSE results yielded for encoding and

reconstructing the specific grinding energy results are a bit different. The specific grinding energy signal has faster frequency components than wheel wear and surface roughness signals. Therefore, the sampling rate (nc) needed to reconstruct the original signal is higher.

On the other hand, the reconstruction of the original in the three signals with less number of carrier waves (nc=20) and less number of points per carrier wave (npc=40) is quite good. Besides, with lower nc and npc values the number of points in the spike train decreases. Therefore, depending of the application field, required accuracy and/or number of training samples, in some cases it is worth using lower nc and npc. Finally, results with npc=20 are by far the worst results for the three signals (wheel wear, surface roughness and specific grinding energy) and not recommended for encoding these analog signals.

9 FUTURE WORK

The present work has highlighted three main lines for future investigation. These lines include the following:

- As stated in Section 4.2.3, for developing the soft sensor and the off-line prediction of the specific grinding energy non-extremely demanding surface finish has been selected as application field. However, there are other application fields widely used in the industry. Therefore, it would be interesting to extend the RNN methodology presented in this work to other application field with different grinding wheels (not only aluminum oxide grinding wheels) and operation conditions.
- The second investigation line is closely related with the previous one. As said in the conclusions of the Chapter 5, further analysis should be carried out with an original database with experiments from different application fields in order to draw better conclusions about the improvement in the downsize of the original database using FCM with weighted inputs.
- The third investigation line could be the most interesting and changeling one. In order to use the Spiking Neural Networks for modelling dynamic evolutions, in this work the first step is presented: encoding analog signals into spikes and, also, recovering the analog signal from a spike train. Once the data is encoded, the other main task is to train the SNN in order to be able to learn the nonlinear relationships between the input and output of the network. Several training algorithms can be found to train the SNN. Researchers developed supervised training algorithm similar to the training algorithm used in Artificial Neural Networks such as SpikeProb, SpikeLM, ReSuMe, Chonotron or SPAN. With these training algorithms the gap between the SNNs and ANNs is minimized. Thus, it could be possible to use the rich theory available for ANNs. However, the SpikeProb and SpikeLM are limited to a single spike per neuron and are not easy to implement while the ReSuMe, Chonotron or SPAN are limited to a single neuron layer. Hence, the next step will be to train a spiking multi-layer network with a backpropagation method with minor changes and using the real data to calculate the error. In fact, using the encoding/decoding algorithm presented in this work it is quite easy to calculate the real error across the network. This can lead

researchers to use Spiking neurons for forecasting and also for deep learning on fields such as Natural Language Processing or Video Recognition.

10 REFERENCES

Aguiar, P., Cruz, C. & Paula, W. C. F., 2008. Predicting Surface Roughness in Grinding using Neural Networks. In: *Advances in Robotics, Automation and Control.* Vienna: I-Tech, pp. 472-484.

Ali, Y. & Zhang, L., 1999 . Surface roughness prediction of ground components using a fuzzy logic approach. *Journal of Materials Processing Technology 89-90,* pp. 561-568.

Anjos, O. et al., 2015. Neural networks applied to discriminate botanical origin of honeys. *Food Chemistry 175,* p. 128–136.

Arab, M. et al., 2014. Micro-controlled Pulse Width Modulator Inverter for Renewable Energy Generators. *Energy Procedia 50,* pp. 832-840.

Balasko, B., Abonyi, J. & Feil, B., 2005. *Fuzzy Clustering and Data Analysis Toolbox (Computer Software).* Veszprem: Department of Process Engineering, University of Veszprem.

Bataineh, K., Naji, M. & Saqer, M., 2011. A Comparison Study between Various Fuzzy Clustering Algorithms. *Jordan Journal of Mechanical and Industrial Engineering 5(4),* pp. 335-343.

Baudry, M., 1998. Synaptic Plasticity and Learning and Memory: 15 Years of Progress. *Neurobiology of Learning and Memory,* pp. 70:113-118.

Beale, M., Hagan, M. & Demuth, H., 2012. *Neural Network ToolboxTM User's Guide.* Natick(MA): The MathWorks Inc..

Bezdek, J., 1981. *Pattern recognition with fuzzy objective function algorithm.* New York: Plenum Press.

Bishop, C., 2006. *Pattern Recognition and Machine Learning.* New York: Springer-Verlag New York.

Bohtea, S., Koka, J. & Han La Poutré, H., 2002. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing 48,* pp. 17-37.

Booji, O. & Nguyen, H., 2005. A gradient descent rule for spiking neurons emitting multiple spikes. *Information Processing Letters 95,* pp. 552-558.

Chae, Y., Horesh, R., Hwang, Y. & Lee, Y., 2016. Artificial neural network model for forecasting sub-hourly electricity usage in commercial buildings. *Energy and Buildings,* Volume 111, pp. 184-194.

Chandana Prasad, P. & Beg, A., 2009. Investigating data preprocessing methods for circuit complexity models. *Expert Systems with Applications,* Volume 36, p. 519–526.

Chandrasekaran, M. & Devarasiddappa, D., 2014. Artificial neural network modeling for surface roughness prediction in cylindrical grinding of Al-SiCp metal matrix composites and ANOVA analysis. *Advances in Production Engineering & Management,* pp. (9) 2:59-70.

Chen, X., Rowe, W. B., Mills, B. & Allanson, D. R., 1998. Analysis and simulation of the grinding process. Part IV: Effects of wheel wear. *International Journal of Machine Tools and Manufacture,* pp. Volume 38, Issues 1–2: 41–49.

Claveira, O. & Torra, S., 2014. Forecasting tourism demand to Catalonia: Neural networks vs. time series models. *Economic Modelling,* Volume 36, pp. 220-228.

Cukrowska, E., Trnková, L., Kizek, R. & Havel, J., 2001. Use of artificial neural networks for the evaluation of electrochemical signals of adenine and cytosine in mixtures interfered with hydrogen evolution.. *Journal of Electroanalytical Chemistry,* 503(1-2), pp. 117-124.

Cybenko, G., 1989. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems ,* 2(4), pp. 303-314 .

Dan Foresse, F. & Hagan, M., 1997. *Gauss-Newton approximation to Bayesian learning.* Houston, IEEE, pp. 1930 - 1935.

DANOBAT, n.d. *DANOBAT.* [Online] Available at: http://www.danobatgroup.com/en/grinding-machines/aerospace-applications/mbtg-dantip [Accessed 2016].

Doimak, n.d. *Doimak S.A..* [Online] Available at: http://www.doimak.es/machine/high-production/ [Accessed 2016].

Dreyfus, G., 2005. *Neural Networks: Methodology and Applications.* Berlin: Springer-Verlag Berlin Heidelberg.

Dunn, J., 1973. A fuzzy relative of the Isodata process and its use in detecting compact well-separated clusters. *Journal of Cybernetics 3(3),* pp. 32-57.

Elman, J., 1990. Finding Structure in Time. *Cognitive Science,* Issue 14, pp. 179-211.

European Commission, 2009. *Transforming the tools of production,* Luxembourg: Office for Official Publications of the European Communities.

Faraoun, K. & Boukelif, A., 2007. Neural Networks Learning Improvement using the K-Means Clustering Algorithm to Detect Network Intrusions. *International Journal of Computational Intelligence 3 (2),* pp. 161-168.

Finelli, L. et al., 2008. Synaptic learning rules and sparse coding in a model sensory system. *PLoS Comput Biol 4,* p. 1–18.

Georgia Grinding Wheel Company, Inc., 2008. *Georgia Grinding Wheel.* [Online] Available at: https://www.georgiagrindingwheel.com/grindingwheels_basics.htm#faq3 [Accessed 28 02 2016].

Gerstner, W. & Kistler, W., 2002. *Spiking Neuron Models. Single Neurons, Populations, Plasticity.* s.l.:Cambridge University Press.

Ghosh-Dastidar, S. & Adeli, H., 2009. A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection. *Neural Networks 22,* pp. 1419-1431.

Godarzi, A., Amiri, R., Tataei, A. & Jamasb, T., 2014. Predicting oil price movements: A dynamic Artificial Neural Network approach. *Energy Policy,* Issue 68, pp. 371-382.

Grüning, A. & Bohte, S., 2014. *ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning.* Bruges (Belgium), s.n.

Gütig, R. & Sompolinsky, H., 2009. Time-warp invariant neuronal processing. *PLoS Biology 7,* p. e1000141.

Guyonneau, R., VanRullen, R. & Thorpe, S., 2004. Temporal codes and sparse representations: A key to understanding rapid processing in the visual system. J Physiol Paris 98: 487–497.. *Journal of Physiology-Paris 98(4-6),* p. 487–497.

Hagan, M. & Menhaj, M., 1994. Training feed-forward networks with the Marquardt algorithm. Volume 5, p. 989–993.

Hagan, M. T., Demuth, H. B., Beale, M. H. & De Jesús, O., 2014. *Neural Network Design.* 2nd Edition ed. United States: Martin Hagan.

Hassan, M., Nath, B. & Kirley, M., 2007. A fusion model of HMM, ANN and GA for stock market forecasting. *Expert Systems with Applications,* Volume 33, pp. 171-180.

Haykin, S., 1998. *Neural Networks: A Comprehensive Foundation.* 2nd Edition ed. Upper Saddle River(NJ): Prentice-Hall.

He, Y., Zhang, Y. & Xiang, L., 2005. *Study of Application Model on BP Neural Network Optimized by Fuzzy Clustering.* s.l., s.n., pp. 712-720.

Hicham, A., Mohamed, B. & Abdellah, E., 2012. An improved approach based on fuzzy clustering and Back-Propagation Neural Networks with adaptive learning rate for sales forecasting; Case study of PCB industry. *International Journal of Computer Science 9 (3),* pp. 404-413.

Hochreiter, S. & Schmidhuber, J., 1997. Long Short-Term Memory. *Neural Computation,* 9(8), pp. 1735-1780.

Hochreiter, S. & Schmidhuber, J., 1997. Long Short-Term Memory. *Neural Computation 9 (8),* p. 1735–1780.

Hodgkin, A. & A. F. Huxley, A., 1952. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology,* p. 117(4):500 .

Hosokawa, A., Mashimo, K., Yamada, K. & Ueda, T., 2004. Evaluation of Grinding Wheel Surface by Means of Grinding Sound Discrimination. *JSME International Journal Series C,* Volume 47, pp. 52-58.

Hough, M. et al., 1999. *SPIKER: Analog waveform to digital spiketrain conversion in ATR's artificial brain (cam-brain) project.* Beppu, Japan, s.n.

Huang, J., Padmanabhan, K. & Collins, O., 2011. The sampling theorem with constant amplitude variable width pulses", IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 58, no. 6, pp. 1178-1190, 2011. *IEEE Transactions on Circuits and Systems I: Regular Papers 58(6),* pp. 1178 - 1190.

Isasi, P. & Galván, I., 2004. *Redes de neuronas artificiales. Un enfoque práctico.* Madrid: Pearson educación.

Izhikevich, E., 2003. Simple Model of Spiking Neurons. *IEEE Transactions on Neural Networks,* pp. 14:1569- 1572.

Izhikevich, E., 2006. Polychronization: computation with spikes. *Neural computation,* p. 18(2):245–282.

Jordan, M., 1990. Attractor dynamics and parallelism in a connectionist sequential machine. *Artificial Neural Networks,* pp. 112-127.

Kasabov, N. et al., 2016. Evolving spatio-temporal data machines based on the NeuCube neuromorphic framework: Design methodology and selected applications. *Neural Networks,* pp. 78:1-14.

Kemp, S., Zaradic, P. & Hansen, F., 2007. An approach for determining relative input parameter importance and significance in artificial neural networks. *Ecological Modelling 204,* pp. 326-334.

Khalid, N., Noor, N. & Ariff, N., 2014. Fuzzy c-Means (FCM) for Optic Cup and Disc Segmentation with Morphologic Operation. *Procedia Computer Science 42,* pp. 255-262.

Kline, S. J., 2015. *2015 World Machine-Tool Output and Consumption Survey,* Cincinnati: Gardner Business Media Inc..

Kure Grinding Wheel, 2016. *Kure Grinding Wheel.* [Online] Available at: http://en.kuretoishi.com/

Laboissiere, L., Fernandes, R. & Lage, G., 2015. Maximum and minimum stock price forecasting of Brazilian power distribution based on artificial neural networks. *Applied Soft Computing,* Issue 35, pp. 66-74.

Lezanski, P., 2001. An intelligent system for grinding wheel condition monitoring. *Journal of Materials Processing Technology,* Volume 109, pp. 258-263.

Liao, T., Hua, G., Qu, J. & Blau, P., 2007. Grinding wheel condition monitoring with hidden Markov model-based clustering methods. *Machining Science and Technology,* Volume 10, p. 511–538.

Liao, T., Hua, G., Qu, J. & Blau, P., 2008. Grinding wheel condition monitoring with boosted minimum distance classifiers. *Mechanical Systems and Signal Processing,* Volume 22, p. 217–232.

Li, G. & Liu, J., 2011. *On-line prediction of surface roughness in cylindrical traverse grinding based on BP+GA algorithm.* Hohhot, IEEE, p. 1456–1459.

Linke, B., 2015. Review on Grinding Tool Wear With Regard to Sustainability. *Journal of Manufacturing Science and Engineering,* 137(6), pp. 060801-060801-8.

Liu, H., Tian, H., X., L. & Li, Y., 2015. Wind speed forecasting approach using secondary decomposition algorithm and Elman neural networks. *Applied Energy,* Issue 157, pp. 183-194.

MacKay, D., 1992. A practical Bayesian framework for backpropagation networks. *Neural Computation,* 4(3), p. 448–472.

Maksoud, T. & Atia, M., 2003. Applications of artificial intelligence to grinding operations via neural networks. *Machining Science and Technology,* Volume 7, p. 361–387.

Malkin, S. & Guo, C., 2008. *Grinding Technology and Applications of Machining with Abrasives.* New York: Industrial Press Inc..

Mao, S., 2014. *Intelligent Utility.* [Online] Available at: http://intelligentutility.com/blog [Accessed 2016].

Marinescu, I. D. et al., 2006. *Handbook of Machining with Grinding Wheels.* s.l.:CRC Press.

Marinescu, I., Rowe, W., Dimitrov, B. & Inasaki, I., 2004. *Tribology of Abrasive Machining Processes.* Norwich(NY): William Andrew Pub..

Mass, W., Natschläger, T. & Markram, H., 2002. Real-Time Computing Without Stable States: A New Framework for Neural Computation Based on Perturbations. *Neural Computing,* 14(11), pp. 2531-2560.

Mehrotra, K., Mohan, C. K. & Ranka, S., 1997. *Elements of Artificial Neural Networks.* Cambridge(MA): MIT Press.

Meng, X. et al., 2016. Pulse width modulation for multi-agent systems. *Automatica 70,* pp. 173-178.

Mohemmed, A., Schliebs, S., Matsuda, S. & Kasabov, N., 2012. SPAN: Spike Pattern Association Neuron for Learning Spatio-Temporal Spike Patterns. *International Journal of Neural Systems 22(4),* p. 1250012.

Nanda, S., Mahanty, B. & Tiwari, M., 2010. Clustering Indian stock market data for portfolio management. *Expert System with Applications 37,* pp. 8793-8798.

Nandi, A. & Pratihar, D., 2004. Design of a genetic-fuzzy system to predict surface finish and power requirement in grinding. *Fuzzy Stets Systems,* Volume 148, pp. 487-504.

Nandi, A. & Pratihar, D., 2004. Design of a genetic-fuzzy system to predict surface finish and power requirement in grinding. *Fuzzy Sets and Systems 148,* pp. 487-504.

Nguyen, D. & Widrow, B., 1990. *Improving the Learning Speed of 2-Layer Neural Networks by Choosing Initial Values of the Adaptive Weights.* San Diego, IEEE, pp. 21-26.

Nikolic, D., Haeusler, S., Singer, W. & Maass, W., 2009. Distributed fading memory for stimulus properties in the primary visual cortex. *PLoS Biology 7,* p. 1–19.

Olden, J. & Jackson, D., 2002. Illuminating the black box: a randomization approach for understanding variable contributions in artificial neural networks. *Ecological Modelling 154,* p. 135–150.

Paliwal, M. & Kumar, U., 2011. Assessing the contribution of variables in feed forward neural network. *Applied Soft Computing 11,* p. 3690–3696.

Perrinet, L. & Samuelides, M., 2002. *Sparse image coding using an asynchronous spiking neural network.* s.l., s.n., p. 313–318.

Philip, N., 2009. *What is there in a training sample?.* Coimbatore, IEEE, pp. 1507-1511.

Philip, N., 2009. *What is there in a training sample?.* s.l., s.n., pp. 1507-1511.

Pimentel, B. & de Souza, R., 2013. A multivariate fuzzy c-means method. *Applied Soft Computing 13,* pp. 1592-1607.

Ponulak, F. & Kasiínski, A., 2010. Supervised learning in spiking neural networks with ReSuMe: sequence learning, classification, and spike shifting. *Neural Computing,* p. 22(2):467–510.

Ponulak, F. & Kasinski, A., 2011. Introduction to spiking neural networks: Information processing, learning and applications. *Acta Neurobiologiae Experimentals,* pp. 71:409-433.

Prabhu, S., Uma, M. & Vinayagam, B., 2015. Surface roughness prediction using Taguchi-fuzzy logic-neural network analysis for CNT nanofluids based grinding process. *Neural Computing & Applications,* pp. 26:41-55.

Reid, D., Hussain, A. & Tawfik, H., 2014. Financial Time Series Prediction Using Spiking Neural Networks. *PLoS ONE 9(8),* p. e103656.

Reid, D., Tawfik, H. & Hussain, A., 2015. *Forecasting Weather Signals Using a Polychronous Spiking Neural Network.* Fuzhou, China, s.n., pp. 116-123.

Ren, C. et al., 2014. Optimal parameters selection for BP neural network based on particle swarm optimization: A case study of wind speed forecasting. *Knowledge-Based Systems 56,* pp. 226-239.

Rowe, W. B., 2009. *Principles of Modern Grinding Technology.* Burlington, Massachusetts: Elsevier Inc..

Schrauwen, B. & Van Campenhout, J., 2003. *BSA, a fast and accurate spike train encoding scheme.* New York, s.n., p. 2825–2830.

Sedighi, M. & Afshari, D., 2010. Creep feed grinding optimization by an integrated GA-NN system. *J. Intell. Manuf. ,* Volume 21, p. 657–663.

Shannon, C., 1949. Communications in the presence of noise. *Proceedings of the IRE 37(1),* pp. 10 - 21.

Shau, H., Mahapatra, S. & Panigrahi, D., 2012. Fuzzy c-means clustering approach for classification of Indian coal seams with respect to their spontaneous combustion susceptibility. *Fuel Processing Technology 104,* pp. 115-120.

Shin, J. et al., 2010. Recognition of partially occluded and totated images with a network of spiking neurons. *IEEE Trans Neural Networks 21,* pp. 1697-1709.

Song, S., Miller, K. & Abbott, L., 2000. Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nature Neuroscience,* p. 3(9):919–926.

Štěpnička, M., Cortez, P., Donate, J. & Štěpničková, L., 2013. Forecasting seasonal time series with computational intelligence. On recent methods and the potential of their combinations.. *Expert Systems with Applications,* 40(6), pp. 1981-1992.

Teufel, E., Kletting, M. & Teich, G., 2003. Modelling the glucose metabolism with backpropagation through time trained Elman nets. *2003 IEEE 13th Workshop on Neural Networks for Signal Processing*, 17-19 September, pp. 789 - 798.

Thorpe, S., Delorme, A. & VanRullen, R., 2001. Spike-based strategies for rapid processing. *Neural Networks 14,* pp. 715-726.

Tian, Z. & Zuo, M., 2010. Health Condition Prediction of Gears Using a Recurrent Neural Network Approach. *IEEE Transactions on Realiability,* Volume 59, pp. 700-705.

Ticknor, J., 2013. A Bayesian regularized artificial neural network for stock market forecasting. *Expert system with Applications,* Volume 40, pp. 5501-5506.

Trauwaert, E., 1988. On the Meaning of Dunn's Partition Coefficient for Fuzzy Clusters. *Fuzzy Sets and Systems 25,* p. 217–242.

Tunstall, H., 2009. Myths vs. Facts: The Realities of Grinding Aerospace Components. *Aerospace Manufacturing and Design.*

Vaz, A., Elsinga, B., van Sark, W. & Brito, M., 2016. An artificial neural network to assess the impact of neighbouring photovoltaics in power forecasting in Utrecht, the Netherlands. *Renewable Energy,* Issue 85, pp. 631-641.

Wang, W. & Zhang, Y., 2007. On fuzzy cluster validity indices. *Fuzzy Sets and Systems 158,* pp. 2095-2117.

Wang, Y. et al., 2015. Artificial neural networks for infectious diarrhea prediction using meteorological factors in Shanghai (China). *Applied soft Computing,* Volume 35, pp. 280-290.

Werbos, P., 1990. Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE ,* 78(10), pp. 1550 - 1560.

Wu, G. & and Lo, S., 2010. Effects of data normalization and inherent-factor on decision of optimal coagulant dosage in water treatment by artificial neural network. *Expert Systems with Applications,* 37(7), pp. 4974-4983.

Yang, Q. & Jin, J., 2010. *Study on Machining Prediction in Plane Grinding based on Artificial Neural Network.* Hangzhou, IEEE, p. 15–16 .

Yang, Z. & Yu, Z., 2012. Grinding wheel wear monitoring based on wavelet analysis and support vector machine. *The International Journal of Advanced Manufacturing Technology,* Volume 62, p. 107–121.

Young II, W., Holland, W. & Weckman, G., 2008. Determining Hall of Fame Status for Major League Baseball Using an Artificial Neural Network. *Journal of Quantitative Analysis in Sports 4 (4).*

Zahid, N., Limouri, M. & Essaid, A., 1999. A new cluster-validity for fuzzy clustering. *Pattern Recognition 32 ,* p. 1089–1097.

Zelinski, P., 2013. *Labor Savings in Grinding.* s.l.:s.n.

Zhang, N., 2011. *Urban Stormwater Runoff Prediction Using Recurrent Neural Networks.* Guilin, Springer Berlin Heidelberg, pp. 610-619.

Zhou, Y. & Wu, Y., 2011. *Analyses on Influence of Training Data Set to Neural Network Supervised Learning Performance.* s.l., s.n., pp. 19-25.

Zhou, Z. et al., 2013. Rice plant-hopper infestation detection and classification algorithms based on fractal dimension values and fuzzy C-means. *Mathematical and Computer Modelling 58,* pp. 701-709.

Zhuo, L. et al., 2014. An SA–GA–BP neural network-based colour correction algorithm for TCM tongue images. *Neurocomputing 134,* pp. 111-116.

eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea