OPEN ACCESS
International Journal of Neural Systems, Vol. 00, No. 0 (2021) 2150033 (18 pages)
© The Author(s)
DOI: 10.1142/S0129065721500337

Hysteresis Modeling in Iron-Dominated Magnets Based on a Multi-Layered Narx Neural Network Approach

Maria Amodeo*,†,‡, Pasquale Arpaia†,‡,¶,
Marco Buzio‡, Vincenzo Di Capua†,‡ and Francesco Donnarumma§
*Department of Electronics and Telecommunications (DET)
Polytechnic University of Turin, Turin 10129, Italy
†Instrumentation and Measurement Laboratory
for Particle Accelerator Laboratory (IMPALab)
Department of Electrical Engineering and
Information Technology (DIETI)
University of Naples Federico II, Naples 80100, Italy
‡Technology Department, CERN – European Organization
for Nuclear Research, 1211 Meyrin, Switzerland
§Institute of Cognitive Sciences and
Technologies (ISTC), National Research Council (CNR)
Via San Martino della Battaglia, 44, Rome 00185, Italy
¶Pasquale.Arpaia@cern.ch; pasquale.arpaia@unina.it

Received 10 May 2021
Accepted 11 May 2021
Published Online 22 July 2021

A full-fledged neural network modeling, based on a *Multi-layered Nonlinear Autoregressive Exogenous Neural Network* (NARX) architecture, is proposed for quasi-static and dynamic hysteresis loops, one of the most challenging topics for computational magnetism. This modeling approach overcomes drawbacks in attaining better than percent-level accuracy of classical and recent approaches for accelerator magnets, that combine hybridization of standard hysteretic models and neural network architectures. By means of an incremental procedure, different Deep Neural Network Architectures are selected, fine-tuned and tested in order to predict magnetic hysteresis in the context of electromagnets. Tests and results show that the proposed NARX architecture best fits the measured magnetic field behavior of a reference quadrupole at CERN. In particular, the proposed modeling framework leads to a percent error below 0.02% for the magnetic field prediction, thus outperforming state of the art approaches and paving a very promising way for future real time applications.

Keywords: Magnetic measurements; Multi-layered NARX; deep networks; ferromagnetic hysteresis; model selection.

1. Introduction

Modeling of quasi-static and dynamic hysteresis loops is one of the most challenging topics in computational magnetism, mainly due to the strong nonlinearity and history dependency shown by ferromagnetic materials.[1–7] This general problem is commonly addressed in literature in the context of electrical machines, which are excited by sinusoidal

¶Corresponding author.

current waveforms.[8–10] Conversely, more complex excitation current waveforms $I(t)$, used in particle accelerators and other magnetic devices operating cyclically, are still an open focus of scientific interest. Such waveforms include quasi-periodic sequences of trapezoidal-shaped pulses, with widely varying slopes (i.e. current or, equivalently, field ramp-rates) and flat-top levels. Flat-tops and flat-bottoms correspond to reversal points of the hysteresis loop and their sequence largely determines the relationship $B(I)$ between current and magnetic field. Under these conditions, $B(I)$ becomes much more complex and hard to predict.[11,12]

In particle accelerators, the beam is accelerated by radio frequency cavities while circulating around a ring made by magnets, which generate a bending field, increasing in proportion to the beam momentum. Accurate knowledge of the magnetic field $B(t)$ at any given time during a magnetic cycle is therefore critical for longitudinal and transversal beam control, power supply control, various beam diagnostics and qualitative feedback to operators. The required accuracy is typically 0.01%.[13] In a restricted number of cases, conventional mathematical models can express the $B(I)$ relationship adequately well. An example is the semi-empirical model of the superconducting bending dipoles of the Large Hadron Collider, which generate a very high field (8.4 T), relatively unaffected by perturbations.[14] In the vastly more common case of iron-dominated magnets, effects due to magnetic saturation, hysteresis and eddy currents may be as large as several percent or more, and the problem becomes orders of magnitude more difficult.[11,12] For example, recent attempts using the well-known Preisach models[15,16] could not attain better than 0.2% accuracy. Also other classes of methods, such as Jiles–Atherton differential models,[17] ultimately turn out to be unsuitable, due to their well-known difficulties in handling minor hysteresis loops. As an alternative, real-time measurements can sometimes be carried out in a suitably equipped reference magnet, either a part of the accelerator ring or powered in series with it. At CERN, six of the synchrotrons function thanks to feedback from such measurements, provided by systems known as “B-trains” (currently in the process of being renovated).[18] In general, however, this kind of real-time measurement systems are complex, expensive and sometimes very impractical to deploy, for example owing to the lack of space for sensors close to the beam vacuum chamber. As a result, there is a strong incentive to investigate novel kinds of models in order to complement or even replace measurements. In addition, even where real-time measurement systems are already implemented, such models may serve as a useful complementary role, for example during periods of hardware maintenance.

Recently, more attention has been directed towards Artificial Neural Networks (ANNs), today used with spectacular results in a variety of domains related to time-series prediction,[19–42] but still relatively unexplored in magnetic applications. In Ref. 43, a hybrid Preisach-Neural Network model is proposed to predict the dynamic hysteresis in ARMCO pure iron, reaching a Normalized Root Mean Square Error of the order of 0.7%. ANN techniques are being used more and more often to model magnetic hysteresis in combination with classical approaches like Preisach, Wlodarski, Chua–Stronsmoe and Jiles–Atherton models.[8,10,44] In Refs. 45 and 46, a different hybrid perspective is proposed by combining an ANN with a Fourier Descriptor to evaluate simulated hysteresis loops. An interesting modeling approach is reported in Ref. 47: an NN approach for modeling dynamic hysteresis is proposed by combining an array of NNs where each NN is dedicated to a particular fixed portion of the dynamic hysteresis loop. The whole hysteretic path is built by the composition of the evaluations made by different NNs. The authors simulate the behavior of a synthetic material with a Jiles–Atherton model and use simulated excitation curves (e.g. sinusoidal waveforms) while the aim of our paper is to rely on real magnetic cycles with a richer frequency content in it. In Ref. 10, the authors study the hysteresis behavior of a transformer core. Their focus is on the prediction of a single hysteresis cycle, and in general they are not able to take into account the magnetic field response over a long period. Moreover, even in the short term case, their accuracy does not meet the above accuracy requirements of 0.01%.[8,10,16,45] In this paper, a different approach is proposed, based on tuning a Multi-layered neural network to fit directly the magnet response, by avoiding complementary physical models. Different architectures are considered and selected according to a compromise between the accuracy of the field estimation and the level of complexity of the network. We show that a Multi-layered Nonlinear Autoregressive Exogenous Neural Network (NARX), which relies on temporal feedback to capture the underlying physics, was the best-performing architecture. The results of tests carried out on a dedicated experimental setup outperform both traditional and hybrid models, suggesting that this is indeed a very promising approach. The paper is organized as follows. In Sec. 2, the mathematical problem is stated and the proposed NARX architecture described. In Sec. 3, the experimental setup is discussed together with the preparation of the dataset used to train and test the networks. In Sec. 4, the model selection algorithm and the different architectures tested are presented. In Sec. 5, we discuss the results achieved in detail.

2. Problem Statement and Architecture Proposal

Our problem can be formally expressed as the estimation of the unknown output y as a function f of K previous outputs, the input u and H previous inputs:

$$\begin{aligned} y(n) = f(&y(n-1), \ldots, y(n-H), \\ &u(n), u(n-1), \ldots, u(n-K)), \end{aligned} \tag{1}$$

where y is the estimated magnetic field, u is the excitation current and n is a discrete time index. The model relies on two buffers, one of network outputs and another of past observations of the input current.

The network's predictive capability is enhanced by endowing it with two buffers taking into account the previous input and output of the system. These are expected to model dynamic features that are either time-dependent, such as eddy current decay transients, or history-dependent, such as magnetic hysteresis. In practice, we propose to approximate Eq. (1) with a Multi-layered NARX model. Given a NARX network with L layers (Fig. 1), the input can be propagated forward through the layers in the following way. The output values a_j^1 of the A_1 neurons belonging to the first layer $l = 1$ of the network are computed by means of the equations

$$\begin{aligned} a_j^1(n) = \phi^{A_1}\Bigg(&\sum_{h=1}^{H} W_{jh}^{O} y(n-h) \\ &+ \sum_{k=0}^{K} W_{jk}^{I} u(n-k)\Bigg), \end{aligned} \tag{2}$$

where ϕ^{A_1} is the activation function of the neurons of the layer 1, the weights W_{jh}^{O} control the strength of the connections from the hth output to the neuron j of the layer 1 and the weights W_{jk}^{I} control the strength of the connection from the kth input to the neuron j of the layer 1. Similarly, for each successive layer it is possible to compute the output values of each neuron by the equations

$$a_j^l(n) = \phi^{A_l}\left(\sum_{i=0}^{A_l} W_{ji}^{l} a_i^{l-1}(n)\right), \tag{3}$$

where $l \in \{2, \ldots, L\}$ is the layer index, ϕ^{A_l} is the activation function of the neurons of the lth layer,

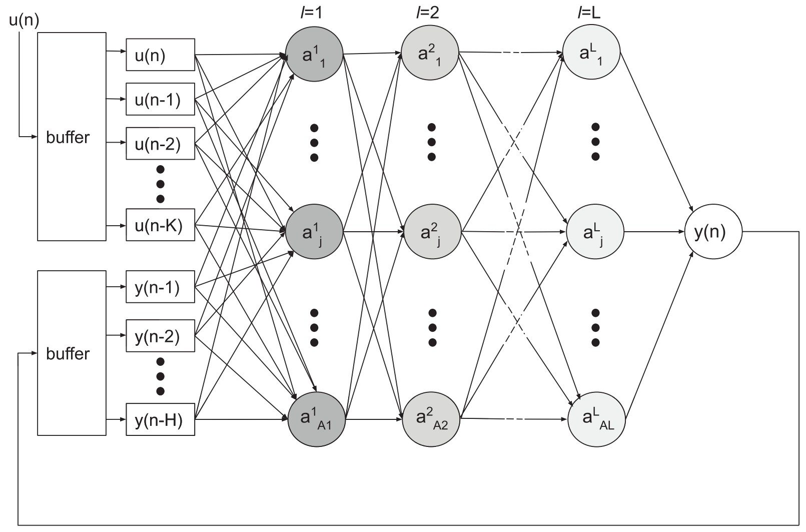


Fig. 1. Multi-layered NARX scheme. The computation of the states a_j^l of the network is forward, starting from first layer (Eq. (2)), then updating internal layers (Eq. (3)), and finally computing the output $y(n)$ (Eq. (4)).

and W^l_{ji} are the weights that control the strength of the connection from the neuron i of the layer $l-1$ to the neuron j of the layer l. We use the conventional formulation in which we set $a^{l-1}_0 = 1$ in order to simulate the bias term W^l_{j0}. The value of the output neuron y is computed as

$$y(n) = \sum_{i=0}^{A_{L-1}} W^E_i a^L_i(n), \tag{4}$$

where we use the identity as activation function, while the weights W^E_i control the strength of the connection coming from the neuron i of the layer L to the output neuron y. In our simulation we set the activation function $\phi^{A_l}(x) = \tanh(x)$ for all the layers $l \in \{1, L\}$. In summary, our architecture is defined by an array of hyperparameters $\theta = (L, \mathbf{A}, K, H)$.

2.1. *Multi-layered NARX literature background*

Since the proof of universal approximation for Feed forward[48] and recurrent[49] neural networks (with the sufficient condition of one hidden layer) the majority of the NN approaches focused on developing networks *in width*, almost neglecting the benefits of developing layers *in depth*. However, an astonishing improvement of NN based system performances was achieved when the possibility of expanding layers in depth became computationally tractable thanks to the development of new smart methods for learning and the increased computational power of computer machines (see Ref. 50 for a comprehensive historical review). Further, when dealing with time series, successful dynamic approaches unfolding the depth of the network *through time* were proposed, like Long-Short Memory Networks (LTSM) and variants.[51–55] In this paper, the proposed modeling choice of a Multi-layered NARX network lies at the edge between a static and a dynamic deep network approach. In general, successful NARX based approaches have been extensively studied (see, e.g. Refs. 35 and 36). NARX models allow a sliding window operation across the feed-forward layers, without relying on recurrent connections. In particular, our modelization takes inspiration from recent 'Jordan' NARX neural network models[56,57] and it is augmented with internal *static* layers. Although in those models the output layer is sent back to the input layers (that is why this kind of modelization is also referred to as *recurrent*), it is worth noting that the computation is completely forward. Consequently, this approach avoids the shortcomings of Backpropagation Through Time (BPTT) learning,[58] which requires unfolding the network through time for as many timesteps as there are in the sequence, which significantly slows down learning and/or causes large memory consumption. Note that the presented formalizations collapse to the definition of Deep MultiLayer Perceptron (MLP) (when $H = K = 0$) and Deep Time Delay Neural Network (TDNN) networks (when $H = 0$). Selecting a model architecture is therefore equivalent to assigning a set of integer values to the hyperparameters $\boldsymbol{\theta}$. The selection process is discussed in detail in Sec. 4.

3. Experimental Setup

3.1. *Measurement setup*

An extensive measurement campaign was performed as a case study on a spare reference quadrupole available at CERN (Fig. 2). The measurement setup is represented schematically in Fig. 3. The magnet was fed by an A&D AG BIP1540 power supply, capable of providing an output current up to 40 A and an output voltage of 10 V. To control the power supply, we used an NI PXI 461 card driven by custom C++ software based on the Flexible Framework for Magnetic Measurements (FFMM).[59] The current was measured with a LEM IT60 ULTRASTAB DCCT having an accuracy of 3 ppm. The magnet was excited with ten different cycle sequences. In order to enforce a specific initial condition (for H and B)

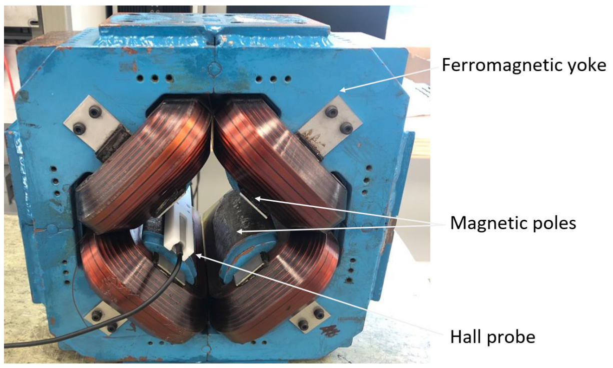


Fig. 2. Reference quadrupole used for the case study. At the tip of the pole, the Hall probe based FM302 teslameter provided by Projekt Elektronic GmbH was used to measure the magnetic field.

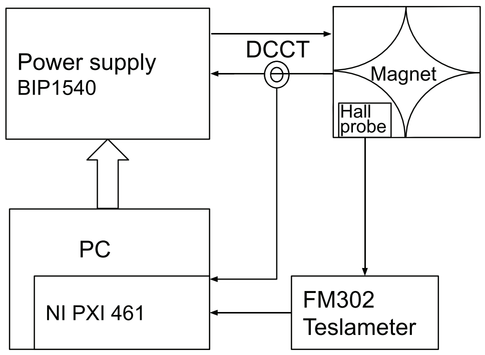


Fig. 3. Schematics of measurement setup.

we decided to perform a degaussing of the magnet before starting our acquisitions.[60] In this way, we are ensuring initial conditions with $H = 0\,\mathrm{A/m}$ and $B = 0$ T. The described shape of the waveforms was designed to use curves similar to the ones actually used in magnets for particle accelerators.[61]

Each sequence includes seven trapezoidal cycles about 4 to 5 seconds long, starting from $I = 0$ and reaching increasingly higher flat-top values, designed to scan the whole interior area of a major hysteresis loop. The major loop corresponds to the maximum applied current of 25 A and each flat-top level represents an inversion point in the magnetic history, which determines the subsequent branch of the hysteresis loop. The flat-top levels in the different sequences were all different, in order to test the interpolating capability of the network. All different levels are listed in Table 1, while a sample subset is plotted in Fig. 4. The field ramp-rate, defined as the slope of the excitation current during the ramps, is kept constant for all training and test cycles, so as to minimize the impact of this variable on the predictive performance of the neural network. This is because the main objective of this paper is modeling the hysteretic part of the response, rather than different dynamics, so we wanted to eliminate as much as possible any confounding variable. The magnetic field was measured with a Hall probe-based FM302 teslameter, from Projekt Elektronic GmbH, with a sensitivity of 1 V/T. The output voltage was acquired with the same PXI card used to control the power supply, with 24 bits of resolution at the raw sampling rate of 2.5 kS/s. The probe was placed at the tip of one of the magnet poles to measure the peak field in the gap (proportional to the quadrupolar field gradient acting as a magnetic lens on the particle beam). The maximum field measured was $B_{\max} = 0.16$ T. The noise level of the measurement, estimated from its standard deviation on the current plateaus, is approximately $13\,\mu$T, i.e. $8.1 \cdot 10^{-5}$ relative to the maximum.

Table 1. Current cycle flat-top values of the 10 cycle sequences tested. The role of each dataset, be it training, validation or test is given in the second column.

Dataset		Cycle index							Unit
Index	Type	1	2	3	4	5	6	7	
1	D_L^{train}	2	5	8	10	15	20	25	A
2	D_L^{train}	1	2	6	8	12	20	25	A
3	D_L^{train}	1	5	8	10	12	21	25	A
4	D_L^{val}	2	4	6	10	13	18	25	A
5	D_L^{test}	3	6	11	16	20	23	25	A
6	D_E, $\bar{D}_{\mathrm{E}}$	3	6	11	16	20	23	25	A
7	D_E, $\bar{D}_{\mathrm{E}}$	2	6	9	13	17	22	25	A
8	D_E, $\bar{D}_{\mathrm{E}}$	3	6	9	15	18	21	25	A
9	D_E, $\bar{D}_{\mathrm{E}}$	1	3	7	9	13	18	25	A
10	D_E, $\bar{D}_{\mathrm{E}}$	2	4	8	11	15	19	25	A

3.2. *Magnet response*

The measured relationship g between the current I and the magnetic field B represents the so called *hysteresis graph* of the magnet:

$$B(I) = g(I). \tag{5}$$

The relationship appears to be essentially linear, although a zoom-in reveals that the field follows a different path when the current is reduced. The area of the hysteresis loop is indicative of the losses, which include a quasi-static contribution intrinsic to the material, plus a dynamic component due to the eddy currents, which increases with the ramp rate. The maximum width of the loop, relative to the full range of the field, is approximately 1% in the region between 7 and 17 A. It is possible to get an insight into the magnetic response, using a Linear Model[62] and in first approximation neglecting its nonlinear part

$$B(I) = B_0 + G \cdot I + \hat{B}(I). \tag{6}$$

Consequently, we can compute $B_0 = 7.79 \cdot 10^{-4}$ T and $G = 6.50 \cdot 10^{-3}$ T/A, respectively, the offset and gain of the least-square linear regression. $\hat{B}(I)$ is the residual of the regression and contains the nonlinear

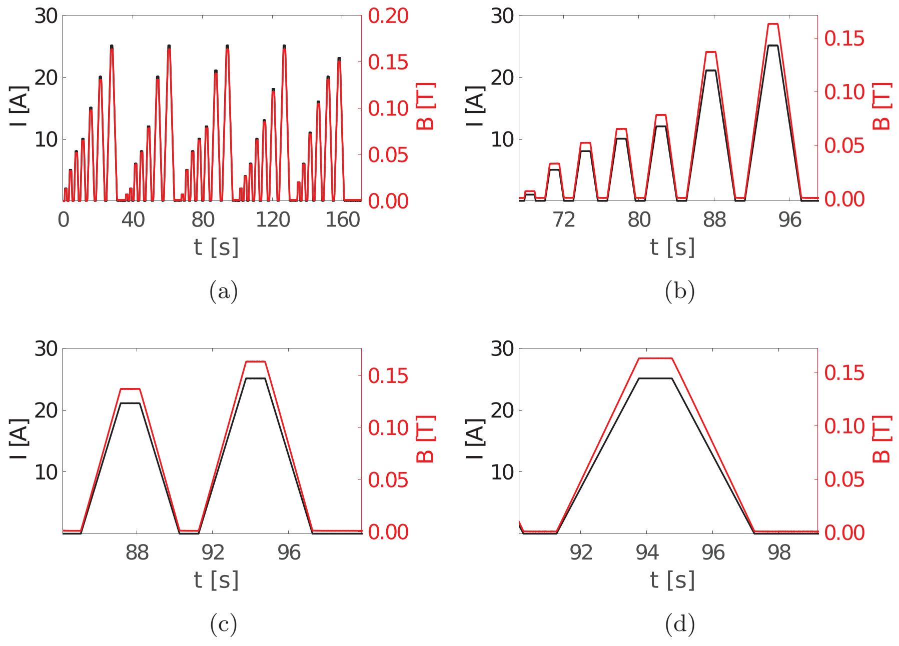


Fig. 4. (Color online) Sample Plot of the current $I(t)$ (in black) and the magnetic field, $B(t)$ (in red) from the collected data. The abscissa represents the time in seconds (obtained knowing the sample rate of the acquisitions of 2.5 kS/s — corresponding to a sampling interval of 0.4 ms). The ordinate has a double scale: on the left for the excitation current $I(t)$ and on the right for the magnetic field $B(t)$. Panel (a) shows the form of a whole sequence of data collected. Panel (b) shows a selection focusing on the first set of cycles, containing seven magnetic cycles; Panels (c) and (d) represent a further zoom on the last two and one magnetic cycle of the set of cycles, respectively.

component. This decomposition is crucial in the construction of datasets used to learn and test models on the nonlinear part of the signal.

A different representation of the magnetic behavior as a function of the current is given in Fig. 5, in terms of the so-called transfer function T_f, defined by the field-to-current ratio

$$T_\mathrm{f}(I) = \frac{B(I)}{I} = \frac{B_0}{I} + G + \frac{\hat{B}(I)}{I}. \tag{7}$$

In this figure, one can more clearly see how the response switches discontinuously from the lower to the upper branch of the hysteresis loop, as the current starts to decrease at the end of each flat-top. The flatness of the lower branch on the up-ramp between about 5 and 25 A corresponds to an almost constant transfer function, i.e. the desired linear behavior under typical operating conditions. Due to the field level being very low, there is no visible saturation leading to a reduction of the transfer function at high current. The nonlinear component $\hat{B}$ vanishes at the high and low reversal points of the hysteresis loops. As a result, the vertical asymptote for $I \to 0$ can be entirely attributed to the remanent

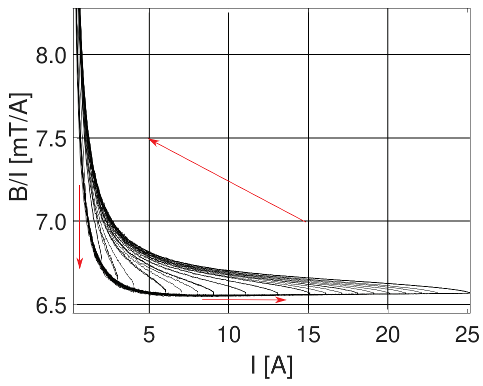


Fig. 5. Transfer function $T_f(I)$ of the magnet, defined by the ratio between magnetic field and excitation current as in Eq. (7).

field B_0. The transfer function B/I is equivalent to the H-B graph and it contains the same amount of information, since in a magnetic circuit the field H is proportional to the excitation current I. In the context of our application, the transfer function is the preferred representation because it allows an operator to visualize more readily the degree of linearity corresponding to a given level of excitation, perfect linearity corresponding of course to a constant transfer function.[63]

3.3. *Dataset preparation*

The raw dataset $\bar{D}$ is composed of the excitation current and field waveforms of the 10 sequences of 7 magnetic cycles, acquired at 2.5 kS/s (sampling time 400 μs) for a total of 871 440 samples. During the network architecture selection and training phases were carried out on a reduced subset D including 87 144 samples at the actual sample rate of 250 S/s (corresponding to a sampling interval of 4 ms.). We used about one half of the subsampled dataset D, D_L, for the architecture learning and the remaining half, D_E, for the test and statistical error evaluation. The data arrays were organized in pairs $[I_L(n), B_L(n)]$ and $[I_E(n), B_E(n)]$ including the measured current and field. The data in D_L was further split into training D_L^{train}, validation D_L^{val} and test D_L^{test} datasets with a 60:20:20 ratio. The splitting of the dataset is detailed in Table 1, where the sequences of flat-top currents are also listed. It should be noted that the training subset includes only a few of the possible transitions between different successive flat-top levels. Each different combination is associated with a different branch of the magnetic hysteresis loop, and the accuracy of the inference made on the test combinations gives a measure of the interpolating power of the trained network. In addition, a second version of the datasets, $\hat{D}_L = [\hat{I}_L(n), \hat{B}_L(n)]$ and $\hat{D}_E = [\hat{I}_E(n), \hat{B}_E(n)]$, was created by replacing the measured field with its nonlinear component, as derived from Eq. (6), $\hat{B}(I) = B(I) - B_0 - G \cdot I$. In this case, the magnetic field can be computed by adding back the NN output $\hat{y}$ to the previously subtracted linear regression. The rationale of this decomposition is to isolate the physically interesting part of the magnet's response, focusing the training process on a dataset having a much smaller dynamic range.

Architectures were simulated within the Neural Network Toolbox in Matlab 2018b. The training and simulations were performed on a computer equipped with an Intel Core i5, Clock 3.2 GHz, Ram 8 GB.

4. Architecture Tuning

4.1. *Model selection and evaluation*

We adopted an incremental approach, by increasing the complexity of the model progressively. First, we considered a static structure without feedback, and we independently optimized the number of layers L, then the number of neurons on each layer A_l. Next, we added feedback first on the input, optimizing K, and then on the output, optimizing H to finally achieve a NARX structure (see Sec. 4.3). The process was carried out on D_L and also on $\hat{D}_L$ as defined in Sec. 3.3. This is because, besides testing the capability of the network architecture of reconstructing the magnetic field ($B \approx y$), we also tested the capability of obtaining B when focusing only on the learning of its nonlinear component and reconstructing it by

Algorithm 1. Model Evaluation and Selection ($\mathcal{M}$, params, D).

```
Require: set $\mathcal{M}$ of hyperparameters $\theta_j$ for each model to evaluate (see Table 3)
    set *params* of simulation parameters (see Table 2)
    the dataset $D$
Ensure: set *Score* of evaluations for each model in $\mathcal{M}$
 1: $[D_L^{\text{train}}, D_L^{\text{val}}, D_L^{\text{test}}] = \text{Split}(D_L, 60:20:20)$
 2: for $\theta_j \in \mathcal{M}$ do                         ▷ Loop1: iterate over the set of different models
 3:     for $i = 1:R$ do                                     ▷ Loop2: repeat learning $R$ times
 4:         repeat
 5:             $W = \text{train}(\theta_j, \text{params}, D_L^{\text{train}}, D_L^{\text{val}})$
 6:         until term_condition (params)                    ▷ Loop3: actual execution of a training instance
 7:         $[B_L^{\text{test}}, I_L^{\text{test}}] \leftarrow D_L^{\text{test}}$
 8:         $y_i \leftarrow y(\theta_j, W; I_L^{\text{test}})$
 9:         $\text{Error}(i) \leftarrow \text{RMSE}(y_i, B^{\text{test}})$          ▷ as defined in Eq. (8)
10:     end for
11:     $\text{Score}(j) = \text{model\_evaluate}(\text{Error}, \theta_j)$          ▷ compute BC scores, Eq. (9)
12: end for
```

adding the linear component (i.e. $B \approx B_0 + Gu + \hat{y}$). The optimal hyperparameters are given in Table 3 and were later used also to evaluate the models trained on the full dataset D_L. The pseudo-code representing each step of the selection process is listed in Algorithm 1, where the input dataset D represents either the full or the nonlinear component only version. The algorithm includes three main loops. The first loop, *Loop*1, iterates over a set of models $\mathcal{M}$, each one defined by its own hyperparameter vector, θ_j. The second loop, *Loop*2, trains the network and estimates its prediction error R times on the test dataset D_L^{test}, in order to improve the statistical significance of the results. The third loop, *Loop*3, is an actual instance of training performed according to the training parameters given in Table 2, and the training and validation datasets, D_L^{train} and D_L^{val}. During the training procedure the hyperparameters of the model θ_j are kept fixed, while the connection weights W are updated iteratively with the objective to minimize the output reconstruction error with the Levenberg–Marquardt (LM) method,[64] until one of the termination criteria is met. These are contained in the function *term_condition* and include:

- the validation error fails to decrease for $maxFail$ iterations,
- the maximum number of epochs for the training (*epochs*) is reached and
- the LM damping factor μ exceeds its maximum acceptable value ($muMax$). In the LM algorithm, the factor μ switches continuously from a Newton-like ($\mu \approx 0$) to a steepest gradient descent ($\mu \gg 0$) optimization. Too large values of μ imply that one is too far from a minimum and the search has failed.

Table 2. Parameters in input to Algorithm 1.

Params	Value	Description
epochs	200	Max. training epochs
maxFail	6	Max. validation failures
minGrad	$1 \cdot 10^{-7}$	Min. gradient
muMax	$1 \cdot 10^{10}$	Max. μ value
R	200	Learning repetitions

Table 3. Hyperparameters definition for the model selection.

Neural network model	Multilayer perception	Time delay	Autoregressive exogenous
Hyperparameter	θ_{MLP}	θ_{TDNN}	θ_{NARX}
L	$\{1,\ldots,15\}$	$\tilde{L}_{\text{MLP}}$	$\tilde{L}_{\text{MLP}}$
$\mathbf{A}$	$\{1,\ldots,10\}^L$	$\tilde{\mathbf{A}}_{\text{MLP}}$	$\tilde{\mathbf{A}}_{\text{MLP}}$
K	0	$\{1,\ldots,35\}$	$\tilde{K}_{\text{TDNN}}$
H	0	0	$\{1,\ldots,35\}$

At the end of the learning phase, the validation dataset D_L^{val} is used to optimize the network generalization. The test dataset D_L^{test} is used to evaluate the prediction performance of the network after the training in terms of the Root Mean Square Error (RMSE), computed for each iteration of *Loop*2 as

$$\text{RMSE}(y_i, D^{\text{test}}) = \sqrt{\frac{\sum\limits_{n\in N}(y_i(n) - B^{\text{test}}(n))^2}{|N|}}, \quad (8)$$

where $y_i = y(\theta_j, W; I_L^{\text{test}})$ is estimated according to Eq. (1) with the current set of hyperparameters and weights, and $|N|$ is the number of samples of the test dataset. To choose the best architecture it is possible to perform a statistical model selection: we maximized the model *evidence* $P(D|\boldsymbol{\theta}_j)$, i.e. a probability term that expresses the preference shown by the data for the jth model of hyperparameters $\boldsymbol{\theta}_j$. In general, the computation of this term is analytically intractable, thus different approximations of this term have been proposed in literature.[62] Popular approximations rely on different penalization terms of the model complexity computed as the number of weights $|W|$ determined by specific choice of hyperparameters θ_j, like Akaike Information Criterion (AIC) or Bayesian Information Criterion (BIC).

For this paper, we rely on a recently proposed information criterion, named Bridge criterion (BC), that aims at bridging the advantages of both AIC and BIC in the asymptotic regime.[65] To this end, the function *model_evaluate* assigns to each hyperparameter θ_j a score based on the BC term

$$\text{Score}(j) = |N| \ln\left(\frac{1}{R}\sum_{i=1}^{R}\text{Error}^2(i)\right) + |N|^{2/3} \cdot (1 + 1/2 + \cdots + 1/|W|). \quad (9)$$

Thus, the minimization of the BC term corresponds to the maximization of the *evidence* of the jth model, ensuring a balanced model fit as the first term in Eq. (9) weighs the reconstruction error, while the second term penalizes the number of weights.

Moreover, a smaller parameter space allows us to find more stable solutions during the training phase.

4.2. *Static Network Structures*

We started by evaluating the performance of a static network, without any feedback. This is the case of a Deep MultiLayer Perceptron (MLP) Neural Network, defined by hyperparameters θ_{MLP} that include the number of hidden layers L, and the number of neurons in each of them, A_l. Following Algorithm 1, we evaluated increasingly complex structures. Figure 6 shows the overall model selection guided by BC scores, in four steps: first two steps for selecting the static structure, last two steps for selecting the input-output buffer size. In the first step, we performed an overall comparison over structures with a fixed number of nodes per layer (10) but a different number of hidden layers (up to 15), aimed at determining the optimal depth for our neural network. In Fig. 6(a), we show BC scores as a function of the number of layers. It is possible to appreciate that networks with a number of layers under 8 gave the best BC scores. After 8 layers the BC scores increase, meaning that the performance of the network models is in general poorer. In Fig. 6(b), the second step of the selection procedure is shown, in which we choose the structures with 2 and 4 layers as suitable candidates for the next selection step, since they provide a good compromise between performance and complexity, and once verified that even adding 8 layers do

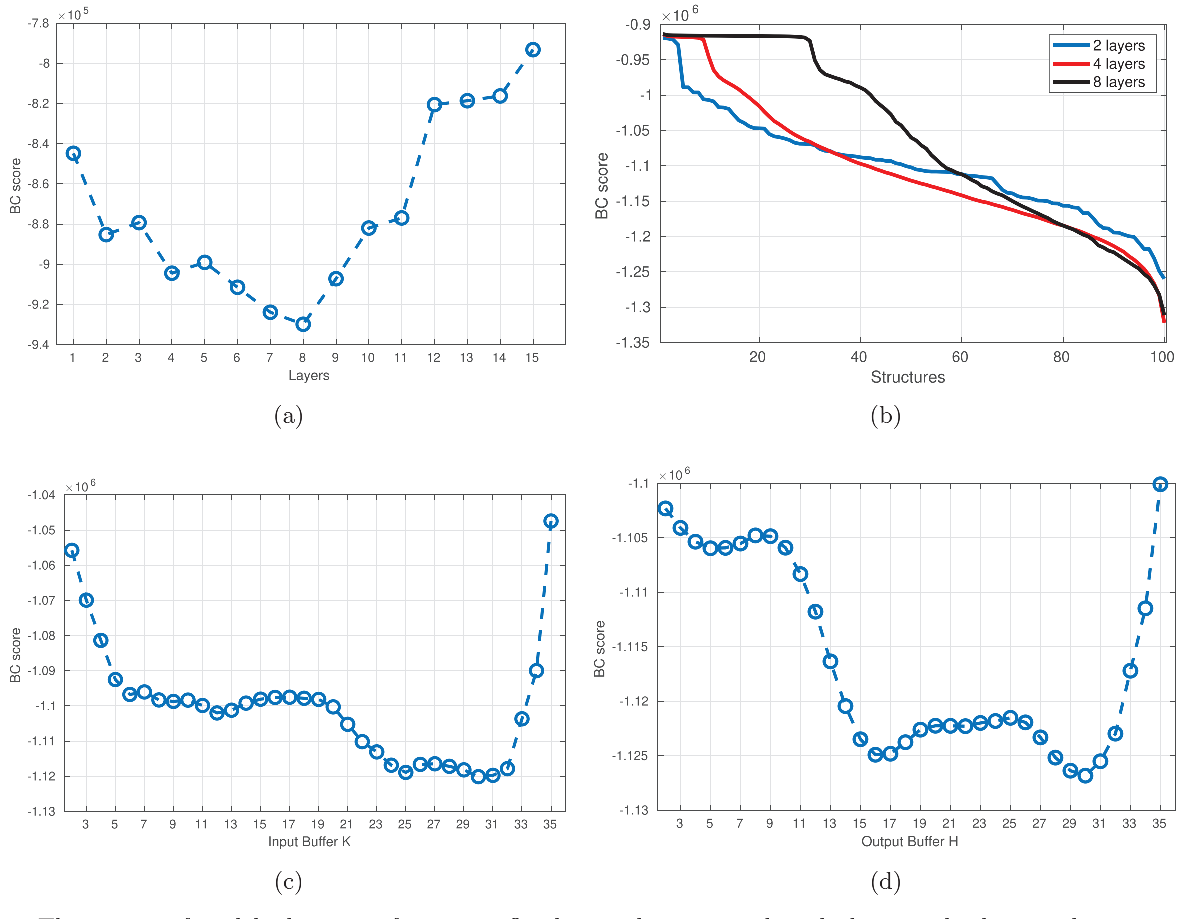


Fig. 6. The process of model selection in four steps. On the top the static and on the bottom the dynamic hyperparameter selection, respectively. Panel (a): firstly, once fixed the number neurons to 10 per layer, we let the number of levels vary and compute BC scores in order to select best promising structures. Panel (b): secondly, fixing the number of the layers, we vary the number of neurons for each layer from 1 to the maximum of 10 neurons, selecting the best 100 winning structures. Panel (c): thirdly, we let the dimension of the input buffer vary. Panel (d): finally, we vary the dimension of the output buffer while all other hyperparameters remain fixed.

Table 4. Hyperparameters $\tilde{\theta}$ selected by Algorithm 1 and used in tests shown in Sec. 5. In the last column, we also list the corresponding number of neuron connections $|W|$.

Hyperparameters	L	$\mathbf{A}$	K	H	$\|W\|$
$\tilde{\theta}_{\text{MLP1}}$	2	$(10, 9)$	0	0	129
$\tilde{\theta}_{\text{MLP2}}$	4	$(1, 1, 1, 10)$	0	0	37
$\tilde{\theta}_{\text{TDNN1}}$	2	$(10, 9)$	26	0	379
$\tilde{\theta}_{\text{TDNN2}}$	4	$(1, 1, 1, 10)$	31	0	67
$\tilde{\theta}_{\text{NARX1}}$	2	$(10, 9)$	26	31	689
$\tilde{\theta}_{\text{NARX2}}$	2	$(7, 8)$	26	17	381
$\tilde{\theta}_{\text{NARX3}}$	4	$(1, 1, 1, 10)$	31	31	98
$\tilde{\theta}_{\text{NARX4}}$	4	$(1, 8, 5, 4)$	31	31	153

not significantly improve BC scores. With these two steps the procedure helps us select static MLP structures with hyperparameters L, $\mathbf{A}$, shown in Table 4.

4.3. *Dynamic network structures*

We augmented the static models with temporal feedback. Our strategy consisted in adding feedback to hidden layers, starting from the best static network structures selected previously. When buffering past observation in input, our model architecture reduces to a deep TDNN with hyperparameters θ_{TDNN}. During this step of model selection, we fixed the best hyperparameters found in the previous section for MLP networks ($\tilde{\theta}_{\text{MLP}}$), while we explored the impact of the input delay buffer length K as an additional hyperparameter. Figures 6(c) and 6(d) show the BC scores as a function of K and H in the range $\{1, \ldots, 35\}$ for the best two- and four-layer structures selected in Sec. 4.2. BC remains low for K between 5 and 33, with a minimum range between 25 and 32. For higher buffer lengths the RMSE increases rather steeply, which indicates instability. Similar behavior can be appreciate for K in the range between 13 and 32, after that performance for higher buffer lengths rapidly degrades. The resulting architecture is a NARX with hyperparameters θ_{NARX}, including the length H of the output buffer. Also in this case, we fine-tuned the model with an incremental approach: we fixed the set of optimal model hyperparameters previously selected (L, $\mathbf{A}$ and K) and only varied H in the range $\{1, \ldots, 35\}$. We found that the best performances are associated with long buffers: in Table 4 the best combinations of hyperparameters $\tilde{\theta}$ selected by Algorithm 1 are shown.

5. Results and Discussion

The performance of these models was evaluated on a completely new collected dataset D_E, which include different sequences of hysteresis inversion points with respect to the learning phase, in order to stress the interpolating power of the networks. First, we compute for each model the estimation $y = y(\theta, W; I_E)$ given by the network on the new dataset. Then, we evaluate the $\text{RMSE}(y, D_E)$ according to Eq. (8). In order to facilitate comparison to the requirements, we normalize the RMSE with respect to the maximum measured field

$$\text{NRMSE}(y, D_E) = \frac{\text{RMSE}(y, D_E)}{B_{\max}} \cdot 100. \qquad (10)$$

We also use two other measures of performance, i.e. the Maximum Absolute Error (MAE)

$$\text{MAE}(y, D_E) = \max\{|y(n) - B_E(n)|\}_{n \in N} \qquad (11)$$

and the Maximum Percent Error (MPE), normalized with respect to the maximum measured field

$$\text{MPE}(y, D_E) = \frac{\text{MAE}(y, D_E)}{B_{\max}} \cdot 100. \qquad (12)$$

The results obtained are summarized in Tables 5 and 6. In Table 5 the test dataset D_E is considered with samples at the same sample rate of the learning dataset 250 S/s, while in Table 6 the $\bar{D}_E$ dataset used for the final testing has been collected at the raw sample rate of 2.5 kS/s, ten time faster than the learning dataset. In both tables we give, for each type of model, the reference to the optimal hyperparameter vector along with the corresponding error norms. The optimal hyperparameters are listed separately in Table 3. The linear regression alone gives a relative error of the order of the percent, which corresponds to the relative width of the hysteresis loop. Such an error, which in other contexts might be taken as an indication of good linearity of the magnet tested, is unacceptable for our application. Next, let us consider the results of the networks trained on the dataset D_L, which are given in the upper half of Tables 5 and 6. Both the static (MLP) and the dynamic networks with input feedback (TDNN) perform as the linear regression alone. The NARX networks, instead, are two orders of magnitude better, achieving a best-case NRMSE of 0.006%. The results evaluated on the reduced dataset are about a factor of two worse, i.e. 0.01%. Thanks to having memory of past outputs, the NARXs are shown to be able

Table 5. Performance comparison among the different architectures, computed on the test datasets D_E at the decimated data rate of 250 S/s. RMSE, NMRSE, MAE and MPE are shown for the Linear Regression alone (LR), the ANNs trained on the full dataset D_L and the hybrid models LR+*, combining LR plus the ANN trained on the nonlinear component only $\hat{D}_L$. The reconstructed magnetic field for models LR+* is computed by Eq. (6), in which the linear component $(B_0 + G \cdot I)$ is computed by means of LR coefficients, while the nonlinear component $(\hat{B})$ is approximated by the corresponding network output. Corresponding hyperparameters are given in Table 4.

Architecture	Test dataset	Hyperparameters	RMSE [T]	NMRSE (%)	MAE [T]	MPE (%)
Linear Regression		G, B_0	$9.07 \cdot 10^{-04}$	$5.67 \cdot 10^{-01}$	$2.60 \cdot 10^{-03}$	1.61
MLP1	D_E	$\tilde{\theta}_{\text{MLP1}}$	$8.70 \cdot 10^{-04}$	$5.44 \cdot 10^{-01}$	$2.60 \cdot 10^{-03}$	1.64
MLP2	D_E	$\tilde{\theta}_{\text{MLP2}}$	$8.83 \cdot 10^{-04}$	$5.52 \cdot 10^{-01}$	$2.50 \cdot 10^{-03}$	1.55
TDNN1	D_E	$\tilde{\theta}_{\text{TDNN1}}$	$7.95 \cdot 10^{-04}$	$4.97 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.21
TDNN2	D_E	$\tilde{\theta}_{\text{TDNN2}}$	$8.05 \cdot 10^{-04}$	$5.03 \cdot 10^{-01}$	$2.20 \cdot 10^{-03}$	1.39
NARX1	D_E	$\tilde{\theta}_{\text{NARX1}}$	$2.12 \cdot 10^{-05}$	$1.32 \cdot 10^{-02}$	$3.36 \cdot 10^{-04}$	$2.10 \cdot 10^{-01}$
NARX2	D_E	$\tilde{\theta}_{\text{NARX2}}$	$2.13 \cdot 10^{-05}$	$1.33 \cdot 10^{-02}$	$3.17 \cdot 10^{-04}$	$1.98 \cdot 10^{-01}$
NARX3	D_E	$\tilde{\theta}_{\text{NARX3}}$	$2.05 \cdot 10^{-05}$	$1.28 \cdot 10^{-02}$	$3.11 \cdot 10^{-04}$	$1.95 \cdot 10^{-01}$
NARX4	D_E	$\tilde{\theta}_{\text{NARX4}}$	$2.05 \cdot 10^{-05}$	$1.28 \cdot 10^{-02}$	$3.12 \cdot 10^{-04}$	$1.95 \cdot 10^{-01}$
LR+MLP1	D_E	G, B_0, $\tilde{\theta}_{\text{MLP1}}$	$8.00 \cdot 10^{-04}$	$5.00 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.16
LR+MLP2	D_E	G, B_0, $\tilde{\theta}_{\text{MLP2}}$	$8.00 \cdot 10^{-04}$	$5.00 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.17
LR+TDNN1	D_E	G, B_0, $\tilde{\theta}_{\text{TDNN1}}$	$8.05 \cdot 10^{-04}$	$5.03 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.21
LR+TDNN2	D_E	G, B_0, $\tilde{\theta}_{\text{TDNN2}}$	$7.97 \cdot 10^{-04}$	$4.98 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.20
LR+NARX1	D_E	G, B_0, $\tilde{\theta}_{\text{NARX1}}$	$7.99 \cdot 10^{-04}$	$4.99 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.19
LR+NARX3	D_E	G, B_0, $\tilde{\theta}_{\text{NARX2}}$	$7.99 \cdot 10^{-04}$	$5.00 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.19
LR+NARX3	D_E	G, B_0, $\tilde{\theta}_{\text{NARX3}}$	$8.00 \cdot 10^{-04}$	$5.00 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.20
LR+NARX4	D_E	G, B_0, $\tilde{\theta}_{\text{NARX4}}$	$8.01 \cdot 10^{-04}$	$5.01 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.19

to reproduce the dynamics of the magnet very accurately. We must remark that the maximum length of the output buffer, $K = 35$, corresponds during the training phase to a total duration of 140 ms, much shorter than the time span necessary to cover even just two consecutive inversion points of the magnetic cycle. While in classical approaches, such as the Preisach models, the complete sequence of inversion points is a necessary input to reconstruct accurately the magnetic history, here we find instead that such information appears to be encoded implicitly by the network, despite the shortness of the output delay buffer. The role of the buffers might therefore be limited to the modeling of short-term dynamics, such as the decay of eddy currents or the ripple of the power supply. This hypothesis seems to be confirmed by the improved performance at the higher sampling rate, corresponding to a buffer duration of 14 ms, which allows a finer modeling on an even shorter time-scale. With the help of Table 7, we show a comparison of our results with respect to those of the state of the art in literature facing similar reconstruction problems. In Ref. 10, the authors used a Deep Neural Network to model the magnetization curve, achieving an RMSE of 0.13%. This result is comparable with our result for an MLP architecture ($8.70 \cdot 10^{-4}$ T), but it is higher than the RMSE obtained from the NARX architecture ($2.12 \cdot 10^{-5}$ T). In Ref. 8, the authors used a Preisach memory block and a feed-forward Neural Network to magnetic hysteresis modeling. The maximum prediction error achieved is around 13% that is higher than our MAE reported in Table 5. In fact, the MAE for a NARX network is of the order of 10^{-4}. In Ref. 16, the author used Preisach to model the hysteretic behavior of a combined magnet, reaching a relative error in the order of 0.2%. In our case, the MPE achieved with a NARX architecture is of about 0.2%.

In Ref. 43, the authors proposed a Preisach-recurrent neural network model to predict the

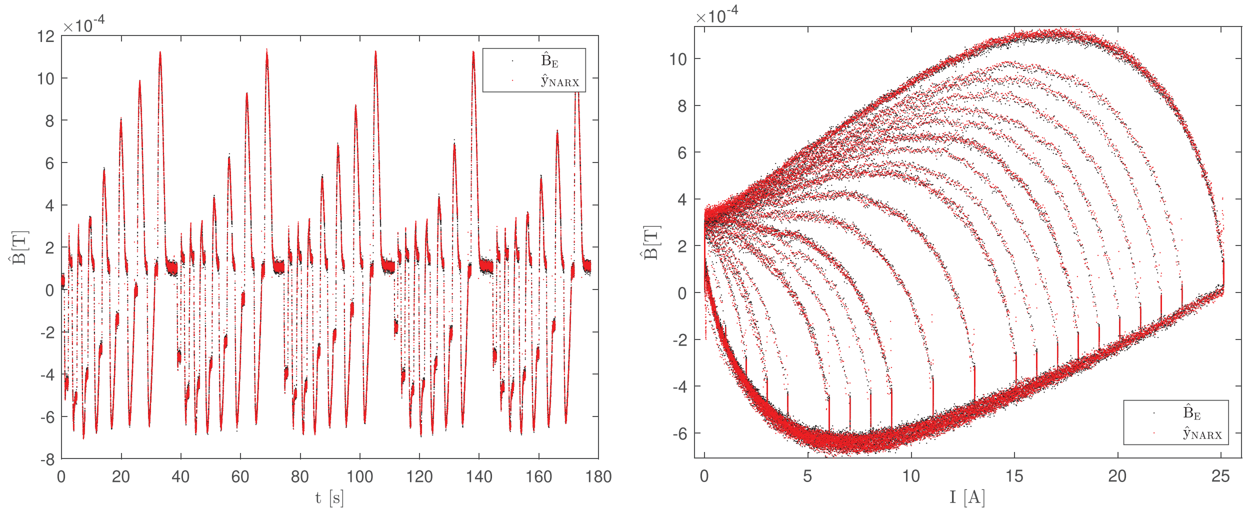
Table 6. Performance comparison among the different architectures, computed on the test dataset $\bar{D}_E$ at the full data rate of 2.5 kS/s. RMSE, NMRSE, MAE and MPE are shown for the Linear Regression alone (LR), the ANNs trained on the full dataset D_L and the hybrid models combining LR plus the ANN trained on the nonlinear component only $\hat{D}_L$. The values of the corresponding hyperparameters are given in Table 4.

Architecture	Test dataset	Hyperparameters	RMSE [T]	NMRSE (%)	MAE [T]	MPE (%)
Linear Regression		G, B_0	$9.07 \cdot 10^{-04}$	$5.67 \cdot 10^{-01}$	$2.60 \cdot 10^{-03}$	1.61
MLP1	$\bar{D}_E$	$\tilde{\theta}_{\mathrm{MLP1}}$	$8.70 \cdot 10^{-04}$	$5.44 \cdot 10^{-01}$	$2.60 \cdot 10^{-03}$	1.64
MLP2	$\bar{D}_E$	$\tilde{\theta}_{\mathrm{MLP2}}$	$8.83 \cdot 10^{-04}$	$5.52 \cdot 10^{-01}$	$2.50 \cdot 10^{-03}$	1.55
TDNN1	$\bar{D}_E$	$\tilde{\theta}_{\mathrm{TDNN1}}$	$1.10 \cdot 10^{-03}$	$6.66 \cdot 10^{-01}$	$2.70 \cdot 10^{-03}$	1.67
TDNN2	$\bar{D}_E$	$\tilde{\theta}_{\mathrm{TDNN2}}$	$8.52 \cdot 10^{-04}$	$5.32 \cdot 10^{-01}$	$2.50 \cdot 10^{-03}$	1.56
NARX1	$\bar{D}_E$	$\tilde{\theta}_{\mathrm{NARX1}}$	$9.92 \cdot 10^{-06}$	$6.20 \cdot 10^{-03}$	$4.63 \cdot 10^{-05}$	$2.89 \cdot 10^{-02}$
NARX2	$\bar{D}_E$	$\tilde{\theta}_{\mathrm{NARX2}}$	$1.27 \cdot 10^{-05}$	$8.00 \cdot 10^{-03}$	$6.90 \cdot 10^{-05}$	$4.31 \cdot 10^{-02}$
NARX3	$\bar{D}_E$	$\tilde{\theta}_{\mathrm{NARX3}}$	$9.22 \cdot 10^{-06}$	$5.80 \cdot 10^{-03}$	$3.98 \cdot 10^{-05}$	$2.49 \cdot 10^{-02}$
NARX4	$\bar{D}_E$	$\tilde{\theta}_{\mathrm{NARX4}}$	$9.28 \cdot 10^{-06}$	$5.80 \cdot 10^{-03}$	$4.06 \cdot 10^{-05}$	$2.54 \cdot 10^{-02}$
LR+MLP1	$\bar{D}_E$	G, B_0, $\tilde{\theta}_{\mathrm{MLP1}}$	$8.00 \cdot 10^{-04}$	$5.00 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.16
LR+MLP2	$\bar{D}_E$	G, B_0, $\tilde{\theta}_{\mathrm{MLP2}}$	$8.00 \cdot 10^{-04}$	$5.00 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.17
LR+TDNN1	$\bar{D}_E$	G, B_0, $\tilde{\theta}_{\mathrm{TDNN1}}$	$8.05 \cdot 10^{-04}$	$5.03 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.18
LR+TDNN2	$\bar{D}_E$	G, B_0, $\tilde{\theta}_{\mathrm{TDNN2}}$	$7.97 \cdot 10^{-04}$	$4.98 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.18
LR+NARX1	$\bar{D}_E$	G, B_0, $\tilde{\theta}_{\mathrm{NARX1}}$	$7.98 \cdot 10^{-04}$	$4.99 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.17
LR+NARX2	$\bar{D}_E$	G, B_0, $\tilde{\theta}_{\mathrm{NARX2}}$	$7.98 \cdot 10^{-04}$	$4.99 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.17
LR+NARX3	$\bar{D}_E$	G, B_0, $\tilde{\theta}_{\mathrm{NARX3}}$	$7.99 \cdot 10^{-04}$	$5.00 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.17
LR+NARX4	$\bar{D}_E$	G, B_0, $\tilde{\theta}_{\mathrm{NARX4}}$	$8.00 \cdot 10^{-04}$	$5.00 \cdot 10^{-01}$	$1.90 \cdot 10^{-03}$	1.17

Table 7. Related results.

Architecture	Metric	Value
Deep Neural Network (MLP with two hidden layers),[10]	Root Mean Square Error	0.13%
Preisach + Feed-forward neural network (one hidden layer),[8]	Maximum Absolute Error	13%
Preisach,[16]	Relative Error	0.2%
Preisach + Recurrent Neural Network,[43]	Normalized Root Mean Square Error	0.7%
Neural Network,[44]	Relative Error	$< 8\%$
Genetic Algorithm + Neural Network,[45]	Mean Square Error	$< 5\%$

dynamic hysteresis in ARMCO pure iron. The proposed model is able to predict the magnetic flux density of ARMCO pure iron with a Normalized Root Mean Square Error (NRMSE) of about 0.7%. If we compare their result with the ones in Table 5, we can see that the NRMSE obtained from a NARX architecture is of the order of 10^{-2}. In Ref. 44, the authors presented a neural network model of nonlinear hysteretic inductors, achieving a relative error less than 8%. In Table 5, the MPE resulting from a NARX architecture is about $2 \cdot 10^{-1}\%$. Finally, in Ref. 45, the authors proposed a combined approach (Genetic Algorithm and Neural Network) to modeling dynamic hysteresis. This approach allows them to achieve a Mean Square Error less than 5%. In our case, in Table 5 the RMSE for the various architectures is shown. In particular, for the NARX architecture, we have an RMSE of the order of 10^{-5}, giving therefore a better result compared to the literature. An example of the nonlinear field component $\hat{B}_E$ (see Sec. 3.1) is plotted as a function of the time or the current, along with the corresponding reconstruction by a NARX network, in Fig. 7. These plots allow to appreciate visually the high quality of the reconstruction, which matches the measured field closely and consistently. Let us now consider the results of the networks trained on the nonlinear component dataset, $\hat{D}_L$, (LR+$*$ group)



(a) Measured vs Estimated Field

(b) Hysteresis graph

Fig. 7. (Color online) Measured ($\hat{B}_E$, in black) and estimated ($\hat{y}_{\mathrm{NARX}}$ with hyperparameters $\tilde{\theta}_{\mathrm{NARX1}}$, in red) nonlinear component of the field $\hat{B}$ in function of time t (Panel (a)) and in function of the current I (Hysteresis graph – Panel (b)).

which are given in the bottom halves of Tables 5 and 6.

This kind of reconstruction can be considered as an hybrid modeling of the magnetic field: a first module corresponding to the linear regression module is coupled with a network which models the nonlinear part only of the signal. In this case, we find a qualitatively different result, since all tested architectures learned on $\hat{D}_L$ perform almost equally and with performance comparable to the MLP and TDNN networks alone, and unable to reach the better performance of NARX learned on the full signal in D_L. These results confirm that avoiding pre-processing at the same time relying complex delay structured in NARX networks is a successful choice to capture the full dynamic of the magnetic field. To perform a statistical evaluation among the models, we performed a one-way analysis of variance (ANOVA) for Absolute Errors in the reconstruction, respectively of Dataset D_E reported in Table 5 and Dataset $\bar{D}_E$ reported in Table 6. A first ANOVA on D_E reconstruction revealed a significant statistical effect on groups, $F[16, 754 \cdot 10^3] = 13 \cdot 10^3$, $p < 10^7$. We also performed post hoc analyses (Tukey's test) that revealed that all models are significantly different from Linear Regression (our null hypothesis). From this analysis, we appreciated that TDNN2 (best performing model excluding NARX models) is not statistically different from MLPs, TDNN1 and LN+TDNN2 ($p > 0.01$). On the other hand, all NARXs are statistically different from all the other models ($p < 10^{-5}$) and are not statistically different from each other ($p > 0.01$). The second ANOVA on $\bar{D}_E$ revealed again a a significant statistical effect on groups, $F(16, 754 \cdot 10^3) = 14 \cdot 10^3$, $p < 10^{-7}$. Post hoc analyses revealed that models augmented with linear regression (LN+$*$) were not significantly different from each other ($p > 0.01$). Moreover, this LR+$*$ group was not statistically different from at least one of the MLP group ($p > 0.01$). This means that at their best they could at most replicate the performance of MLP models and this confirms that the learning restricted to the nonlinear part only does cut off important information of the original signal.

The statistical results on TDNN models are even more interesting: TDNN2 is not statistically different from MLP1 ($p > 0.01$) and TDNN1 is different from all the other models being the worst one, failing to generalize the case of faster sample rate. This is explained by observing that changing the buffer on the input is not sufficient to the TDNN model to let it adapt to signal when different input rates are given. On the other hand, in the case of NARX the buffer on input delay allows the nets to adapt very smoothly to the new faster rate, thanks to the buffer on internal outputs. Thus, all models in NARX group perform better and are statistically different from other groups ($p < 10.6 \cdot 10^{-6}$) and interestingly

Table 8. Training and simulation times.

Architecture	Training time [s]	Simulation time (D_E)[s]	Simulation time ($\bar{D}_E$) [s]
MLP1	16.88	0.40	1.08
MLP2	14.18	0.44	0.88
TDNN1	46.30	1.73	12.50
TDNN2	22.13	1.50	16.16
NARX1	125.52	2.66	25.23
NARX2	42.96	2.30	22.23
NARX3	19.91	2.91	27.98
NARX4	15.89	2.84	28.67

the different NARXs with different selected hyper-parameters are not statistically different ($p > 0.1$). It is possible to visualize the result of this second ANOVA in Fig. 8: here we show the Absolute Error with 95% confidence interval (straight line) for each model. We can partition the models into four groups: linear regression is our baseline (in gray). The only model performing worst of this baseline is TDNN1 (pink group) that completely fails to adapt to the new sample rate of $\bar{D}_E$. Then, the behavior of the models learned on linear models (LN+$*$ group) is equiparable to the MLP networks, performing better than LN alone (yellow group). On the other hand, TDNN2 is slightly better than this group (blue group). And finally, all NARXs (orange group) are successful in adapting to the new rate and significantly outperform the performances of all the other groups. A final test is made computing training and simulation time of execution of the winning architectures and results are shown in Table 8. The first column reports the architectures on which the training and simulation times are evaluated. In particular, for the training/simulation time evaluation we considered the neural networks trained on the dataset D_L. The second column contains the training time for each architecture. The training time refers to the time needed for the function Train (see line 5 in Algorithm 1). The third and the fourth columns contain the simulation time computed on the test dataset D_E at the decimated data rate of 250 S/s (see Table 5) and the dataset $\bar{D}_E$) at full data rate of 2.5 kS/s (see Table 6), respectively. The simulation time refers to the time needed for the evaluation of the predictions on all the points of the dataset (see line 8 in Algorithm 1).

It is possible to appreciate that this result nicely fits the complexity measure in Table 4. The more complex a model is, the more execution time is required to complete the different steps of the Algorithm 1. In accordance with the complexity measure, it is worth mentioning that deeper models of NARX perform better than NARXs with fewer layers. On the other hand, it should be noted that computation occurring on the same layer can be further optimized by processing them in parallel, thus a trade-off can be achieved between those constraints in order to reach the best performance in time execution.

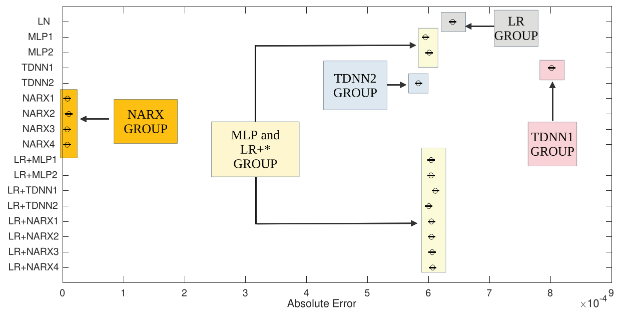


Fig. 8. (Color online) ANOVA results on Absolute Errors computed for the competing models on Dataset $\bar{D}_E$. The horizontal lines are the 95% confidence interval for each model. Five groups are highlighted: LR group (gray), MLP group and LR+$*$ group (yellow), TDNN1 (pink), TDNN2 (in blue) and the NARX group (orange).

6. Summary and Conclusions

We developed an incremental method to select an optimal DNN architecture to predict the field generated by a magnet, when excited by a sequence of cyclic excitation current waveforms. We tested it experimentally on a case study in conditions representative of those found in particle accelerators and similar, pulsed-mode machines. The response of the magnet tested is linear within about 1.5%, and we focused essentially on predicting its residual nonlinear component, which is dominated by ferromagnetic hysteresis. When we trained and tested our network directly on the raw datasets, we found that NARX networks achieve in general the required level of performance i.e. an NRMSE better than 0.01%, while simpler architectures with buffers only on the input (TDNN) or no buffers at all (MLP) do not.

Such excellent performance is well within our initial requirements, and paves a very promising way for future applications in this context. We observed that the prediction accuracy generally improves when the network is trained on low data rate (250 S/s) signals and tested at higher data rate (2.5 kS/s). This may be linked to the fact that the reduced dataset is less affected by noise, and therefore allows the network to better focus on capturing the underlying dynamics. In the near future, we intend to investigate this aspect by decreasing further the data rate of the training dataset, while at the same time increasing the interval between the samples in the output buffer so that it may cover the time span of two or more cycles. In this way, we aim at systematically feeding the network information about the most recent field reversal points in the hysteresis loop, which may further improve the long-term prediction capability. We also plan to train and test NARX networks on a wider variety of excitation waveforms, such as sequences of cycles with flat-tops increasing or decreasing randomly, which are representative of the most challenging actual operating conditions of accelerator magnets. Such tests will also address the concern that the very good results we presented might be biased by overtraining, linked to a specific category of excitation waveforms. In addition, in future experimental campaigns we will extend the range of the tested currents, to ensure the introduction of relevant levels of saturation, as well as the range of current ramp rates in order to deal with different levels of eddy current-related effects.

In this paper, we focused on an incremental approach that firstly optimizes the static structure parameters $(L, \mathbf{A})$ and then the time buffers (i.e. K, H). While this approach does not ensure the finding of an optimal solution, we showed that it constitutes an efficient heuristic able to computationally minimize the model selection procedure. Future work will also focus on extending and refining the model selection by including smart procedures for parameter grid search, trainable activation functions and sparse structure learning (see e.g. Refs. 66–70) that would allow deeper structures to be better managed. Moreover, a further improvement will be to expand the framework in order to include classification capabilities[34,71] to identify different branches of the hysteresis cycles in real time. Overall, in this framework, the analysis of the simpler network solutions found by the procedure could also open to the possibility of producing even more efficient solutions, by substituting blocks of NN operations with equivalent mathematical equations or equivalent smaller networks. Finally, as part of the renovation of the real-time magnetic measurement systems currently ongoing at CERN, we are implementing in FPGA hardware a real-time version of the NARX networks that will be able to carry out a continuous field prediction, in parallel to the measurement. This facility will provide the opportunity to gather huge amounts of data concerning thousands of different sequences of cycles, covering all relevant dynamic scenarios. This will ultimately allow us to fine-tune the parameters of the networks, and estimate their robustness and performance in the long term with high statistical significance.

References

1. A. Visintin, Mathematical models of hysteresis, in *Modelling and Optimization of Distributed Parameter Systems Applications to Engineering* (Springer, 1996), pp. 71–80.
2. V. Hassani, T. Tjahjowidodo and T. N. Do, A survey on hysteresis modeling, identification and control, *Mech. Syst. Sig. Process.* **49**(1–2) (2014) 209–233.
3. I. D. Mayergoyz, *Mathematical Models of Hysteresis and Their Applications* (Academic Press, 2003).
4. E. D. Torre, Hysteresis modeling, *COMPEL* **17**(6) (1998) 682–689.
5. J. Takács, A phenomenological mathematical model of hysteresis, *COMPEL* **20**(4) (2001) 1002–1015.
6. M. F. Jaafar, Magnetic hysteresis modeling and numerical simulation for ferromagnetic materials, in *2013 Int. Conf. Control, Decision and Information Technologies* (*CoDIT*) (IEEE, Hammamet, 2013), pp. 516–523.
7. D. C. Jiles and D. L. Atherton, Theory of ferromagnetic hysteresis, *J. Appl. Phys.* **55**(6) (1984) 2115–2120.
8. C. Serpico and C. Visone, Magnetic hysteresis modeling via feed-forward neural networks, *IEEE Trans. Magn.* **34**(3) (1998) 623–628.
9. D. Makaveev, L. Dupré, M. De Wulf and J. Melkebeek, Modeling of quasistatic magnetic hysteresis with feed-forward neural networks, *J. Appl. Phys.* **89**(11) (2001) 6737–6739.
10. A. Du, J. Pan and A. Guzzomi, Modeling the hysteresis characteristics of transformer core under various excitation level via on-line measurements, *Electronics* **7** (2018) 390.
11. P. Arpaia, M. Buzio, F. Caspers, G. Golluccio and C. Petrone, Static metrological characterization of a ferrimagnetic resonance transducer for real-time magnetic field markers in particle accelerators, in *2011 IEEE Int. Instrumentation and Measurement Technology Conf.* (IEEE, Hangzhou, 2011), pp. 1–4.
12. N. Sammut, L. Bottura and J. Micallef, Mathematical formulation to predict the harmonics of

the superconducting large hadron collider magnets, *Phys. Rev. Accel. Beams* **9**(1) (2006) 012402.

13. E. Feldmeier *et al.*, Magnetic field correction in normal conducting synchrotrons, in *Proc. 1st Int. Particle Accelerator Conf.* (*IPAC'10*) (JACoW Publishing, Kyoto, 2010), pp. 675–677.
14. M. Di Castro, D. Sernelius, L. Bottura, L. Deniau, N. Sammut, S. Sanfilippo and W. V. Delsolaro, Parametric field modeling for the LHC main magnets in operating conditions, in *2007 IEEE Particle Accelerator Conf.* (*PAC*) (IEEE, Albuquerque, NM, 2007), pp. 1586–1588.
15. F. Preisach, Über die magnetische nachwirkung, *Z. Phys.* **94**(5–6) (1935) 277–302.
16. V. Pricop, Efectele histerezisului din materialele folosite pentru circuitele magnetice ale acceleratoarelor de particule, hysteresis effects in the cores of particle accelerator magnets, Ph.D. thesis, Transilvania University of Brasov (2016).
17. D. C. Jiles and D. L. Atherton, Theory of ferromagnetic hysteresis, *J. Magn. Magn. Mater.* **61**(1–2) (1986) 48–60.
18. C. Grech, M. Amodeo, A. Beaumont, M. Buzio, V. Di Capua, D. Giloteaux, N. Sammut and J. V. Wallbank, Error characterization and calibration of real-time magnetic field measurement systems, *Nucl. Instrum. Methods Phys. Res., Sect. A* **990** (2020) 164979.
19. F. Donnarumma, R. Prevete and G. Trautteur, Programming in the brain: A neural network theoretical framework, *Connect. Sci.* **24**(2–3) (2012) 71–90.
20. F. C. Morabito, M. Campolo, N. Mammone, M. Versaci, S. Franceschetti, F. Tagliavini, V. Sofia, D. Fatuzzo, A. Gambardella, A. Labate, L. Mumoli, G. G. Tripodi, S. Gasparini, V. Cianci, C. Sueri, E. Ferlazzo and U. Aguglia, Deep learning representation from electroencephalography of early-stage Creutzfeldt–Jakob disease and features for differentiation from rapidly progressive dementia, *Int. J. Neural Syst.* **27**(2) (2017) 1650039.
21. Y. Zhang, Y. Wang, J. Jin and X. Wang, Sparse bayesian learning for obtaining sparsity of EEG frequency bands based feature vectors in motor imagery classification, *Int. J. Neural Syst.* **27**(2) (2017) 1650032.
22. A. Geminiani, C. Casellato, A. Antonietti, E. D'Angelo and A. Pedrocchi, A multiple-plasticity spiking neural network embedded in a closed-loop control system to model cerebellar pathologies, *Int. J. Neural Syst.* **28**(5) (2018) 1750017.
23. W. Li, M. Li, H. Zhou, G. Chen, J. Jin and F. Duan, A dual stimuli approach combined with convolutional neural network to improve information transfer rate of event-related potential-based brain-computer interface, *Int. J. Neural Syst.* **28**(10) (2018) 1850034.
24. S. Plakias and Y. S. Boutalis, Lyapunov theory-based fusion neural networks for the identification of dynamic nonlinear systems, *Int. J. Neural Syst.* **29**(9) (2019) 17.
25. T. Yang, C. Cappelle, Y. Ruichek and M. El Bagdouri, Multi-object tracking with discriminant correlation filter based deep learning tracker, *Integr. Comput.-Aided Eng.* **26**(3) (2019) 273–284.
26. P. Lara-Benítez, M. Carranza-García, J. García-Gutiérrez and J. C. Riquelme, Asynchronous dual-pipeline deep learning framework for online data stream classification, *Integr. Comput.-Aided Eng.* **27**(2) (2020) 101–119.
27. J. García-González, J. M. Ortiz-de Lazcano-Lobato, R. M. Luque-Baena and E. López-Rubio, Background subtraction by probabilistic modeling of patch features learned by deep autoencoders, *Integr. Comput.-Aided Eng.* **27**(3) (2020) 253–265.
28. S. Hamreras, B. Boucheham, M. A. Molina-Cabello, R. Benitez-Rochel and E. Lopez-Rubio, Content based image retrieval by ensembles of deep learning object classifiers, *Integr. Comput.-Aided Eng.* **27**(3) (2020) 317–331.
29. D. Simões, N. Lau and L. P. Reis, Exploring communication protocols and centralized critics in multi-agent deep learning, *Integr. Comput.-Aided Eng.* **27**(4) (2020) 333–351.
30. D. Diaz-Vico, J. Prada, A. Omari and J. Dorronsoro, Deep support vector neural networks, *Integr. Comput.-Aided Eng.* **27**(4) (2020) 389–402.
31. U. R. Acharya, S. L. Oh, Y. Hagiwara, J. H. Tan, H. Adeli and D. P. Subha, Automated EEG-based screening of depression using deep convolutional neural network, *Comput. Methods Programs Biomed.* **161** (2018) 103–113.
32. G. B. Martins, J. P. Papa and H. Adeli, Deep learning techniques for recommender systems based on collaborative filtering, *Expert Syst.* **37**(6) (2020) e12647.
33. H. S. Nogay and H. Adeli, Detection of epileptic seizure using pretrained deep convolutional neural network and transfer learning, *Eur. Neurol.* **83**(6) (2020) 602–614.
34. M. Ahmadlou and H. Adeli, Enhanced probabilistic neural network with local decision circles: A robust classifier, *Integr. Comput.-Aided Eng.* **17**(3) (2010) 197–210.
35. T. Lin, B. G. Horne, P. Tino and C. L. Giles, Learning long-term dependencies in NARX recurrent neural networks, *IEEE Trans. Neural Netw.* **7**(6) (1996) 1329–1338.
36. H. T. Siegelmann, B. G. Horne and C. L. Giles, Computational capabilities of recurrent NARX neural networks, *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **27**(2) (1997) 208–215.

37. P. Lara Bentez, M. Carranza Garca and J. C. Riquelme, An experimental review on deep learning architectures for time series forecasting, *Int. J. Neural Syst.* **31**(3) (2021) 2130001.
38. A. Lozano, J. S. Surez, C. Soto-Snchez, J. Garrigs, J. J. Martnez-Alvarez, J. M. Ferrndez and E. Ferndez, Neurolight: A deep learning neural interface for cortical visual prostheses, *Int. J. Neural Syst.* **30**(9) (2020) 2050045.
39. A. OShea, R. Ahmed, G. Lightbody, E. Pavlidis, R. Lloyd, F. Pisani, W. Marnane, S. Mathieson, G. Boylan and A. Temko, Deep learning for EEG seizure detection in preterm infants, *Int. J. Neural Syst.* **31**(8) (2021) 2150008.
40. M. Leming, J. M. Grriz and J. Suckling, Ensemble deep learning on large, mixed-site fMRI datasets in autism and other tasks, *Int. J. Neural Syst.* **30**(7) (2020) 2050012.
41. O. Reyes and S. Ventura, Performing multi-target regression via a parameter sharing-based deep network, *Int. J. Neural Syst.* **29**(9) (2019) 1950014.
42. M. Bernert and B. Yvert, An attention-based spiking neural network for unsupervised spike-sorting, *Int. J. Neural Syst.* **29**(8) (2019) 1850059.
43. C. Grech, M. Buzio, M. Pentella and N. Sammut, Dynamic ferromagnetic hysteresis modelling using a Preisach-recurrent neural network model, *Materials* **13**(11) (2020) 2561.
44. S. Cincotti, M. Marchesi and A. Serri, A neural network model of parametric nonlinear hysteretic inductors, *IEEE Trans. Magn.* **34**(5) (1998) 3040–3043.
45. A. Salvini, F. R. Fulginei and C. Coltelli, A neurogenetic and time-frequency approach to macromodeling dynamic hysteresis in the harmonic regime, *IEEE Trans. Magn.* **39**(3) (2003) 1401–1404.
46. A. Salvini, F. R. Fulginei and G. Pucacco, Generalization of the static Preisach model for dynamic hysteresis by a genetic approach, *IEEE Trans. Magn.* **39**(3) (2003) 1353–1356.
47. F. R. Fulginei and A. Salvini, Neural network approach for modelling hysteretic magnetic materials under distorted excitations, *IEEE Trans. Magn.* **48**(2) (2012) 307–310.
48. K. Hornik, M. Stinchcombe and H. White, Multilayer feedforward networks are universal approximators, *Neural Netw.* **2**(5) (1989) 359–366.
49. K. Funahashi and Y. Nakamura, Approximation of dynamical systems by continuous time recurrent neural networks, *Neural Netw.* **6**(6) (1993) 801–806.
50. J. Schmidhuber, Deep learning in neural networks: An overview, *Neural Netw.* **61** (2015) 85–117.
51. S. Hochreiter and J. Schmidhuber, Long short-term memory, *Neural Comput.* **9**(8) (1997) 1735–1780.
52. J. A. Pérez-Ortiz, F. A. Gers, D. Eck and J. Schmidhuber, Kalman filters improve LSTM network performance in problems unsolvable by traditional recurrent nets, *Neural Netw.* **16**(2) (2003) 241–250.
53. F. Donnarumma, R. Prevete, A. de Giorgio, G. Montone and G. Pezzulo, Learning programs is better than learning dynamics: A programmable neural network hierarchical architecture in a multi-task scenario, *Adapt. Behav.* **24**(1) (2016) 27–51.
54. F. Donnarumma, R. Prevete, F. Chersi and G. Pezzulo, A programmer–interpreter neural network architecture for prefrontal cognitive control, *Int. J. Neural Syst.* **25**(6) (2015) 1550017.
55. Y. Li, Z. Yu, Y. Chen, C. Yang, Y. Li, X. Allen Li and B. Li, Automatic seizure detection using fully convolutional nested LSTM, *Int. J. Neural Syst.* **30**(4) (2020) 2050019.
56. H. Wang and G. Song, Innovative NARX recurrent neural network model for ultra-thin shape memory alloy wire, *Neurocomputing* **134** (2014) 289–295.
57. M. S. H. Lipu, M. A. Hannan, A. Hussain, M. H. Saad, A. Ayob and F. Blaabjerg, State of charge estimation for lithium-ion battery using recurrent NARX neural network model based lighting search algorithm, *IEEE Access* **6** (2018) 28150–28161.
58. P. Vlachas, J. Pathak, B. Hunt, T. Sapsis, M. Girvan, E. Ott and P. Koumoutsakos, Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics, *Neural Netw.* **126** (2020) 191–217.
59. P. Arpaia, L. Bottura, M. Buzio, D. Della Ratta, L. Deniau, V. Inglese, G. Spiezia, S. Tiso and L. Walckiers, A software framework for flexible magnetic measurements at CERN, in *2007 IEEE Instrumentation & Measurement Technology Conf. IMTC 2007* (IEEE, Warsaw, 2007), pp. 1–4.
60. M. S. Fabus and S. Peggs, Hysteresis and degaussing of H1 dipole magnets, Technical Report Number(s): CBETA/045; BNL-211964-2019-TECH (2019), doi:10.2172/1556890, https://www.osti.gov/biblio/1556890.
61. S. Gilardoni *et al.*, Fifty years of the cern proton synchrotron, preprint (2013), arXiv:1309.6923.
62. C. M. Bishop, *Pattern Recognition and Machine Learning* (Springer, 2006).
63. W. Venturini Desolaro, L. Bottura, Y. Chaudhari, M. Karppinen and N. Sammut, Measurement and modeling of magnetic hysteresis in the LHC superconducting correctors, *10th Eur. Particle Accelerator Conf.*, 26–30 June 2006, Edinburgh, pp. 2026–2028.
64. S. Roweis, Levenberg–Marquardt optimization, Lecture Notes (University of Toronto, 1996).
65. J. Ding, V. Tarokh and Y. Yang, Model selection techniques: An overview, *IEEE Signal Process. Mag.* **35**(6) (2018) 16–34.
66. P. C. Bhat, H. B. Prosper, S. Sekmen and C. Stewart, Optimizing event selection with the random grid search, *Comput. Phys. Commun.* **228** (2018) 245–257.

67. K. M. R. Alam, N. Siddique and H. Adeli, A dynamic ensemble learning algorithm for neural networks, *Neural Comput. Appl.* **32**(12) (2020) 8675–8690.
68. F. Donnarumma, R. Prevete, D. Maisto, S. Fuscone, E. M. Irvine, M. A. van der Meer, C. Kemere and G. Pezzulo, A framework to identify structured behavioral patterns within rodent spatial trajectories, *Sci. Rep.* **11**(1) (2021) 1–20.
69. P. Arpaia, F. Donnarumma, A. Esposito and M. Parvis, Channel selection for optimal EEG measurement in motor imagery-based brain-computer interfaces, *Int. J. Neural Syst.* **31**(3) (2021) 2150003.
70. A. Apicella, F. Isgr, R. Prevete and G. Tamburrini, Middle-level features for the explanation of classification systems by sparse dictionary methods, *Int. J. Neural Syst.* **30**(8) (2020) 2050040.
71. M. H. Rafiei and H. Adeli, A new neural dynamic classification algorithm, *IEEE Trans. Neural Netw. Learn. Syst.* **28**(12) (2017) 3074–3083.