Received: 2 January 2023 | Revised: 9 June 2023 | Accepted: 5 October 2023

DOI: 10.1002/acs.3701

RESEARCH ARTICLE

WILEY

Data-driven performance metrics for neural network learning

Angelo Alessandri[1] | **Mauro Gaggero**[2] | **Marcello Sanguineti**[2,3]

[1]DIME, University of Genoa, Genoa, Italy

[2]INM, National Research Council of Italy, Genoa, Italy

[3]DIBRIS, University of Genoa, Genoa, Italy

Correspondence
Angelo Alessandri, DIME, University of Genoa, Via Opera Pia 15, I-16145 Genoa, Italy.
Email: angelo.alessandri@unige.it

Funding information
Italian Ministry of Enterprises and Made in Italy, Grant/Award Number: F/310027/01-03/X56; National Research Council of Italy, Grant/Award Number: DIT.AD021.104; Italian Ministry of University and Research, Grant/Award Numbers: ECS00000035, 2022S8XSMY

Summary

Effectiveness of data-driven neural learning in terms of both local mimima trapping and convergence rate is addressed. Such issues are investigated in a case study involving the training of one-hidden-layer feedforward neural networks with the extended Kalman filter, which reduces the search for the optimal network parameters to a state estimation problem, as compared to descent-based methods. In this respect, the performances of the training are assessed by using the Cramér-Rao bound, along with a novel metric based on an empirical criterion to evaluate robustness with respect to local minima trapping. Numerical results are provided to illustrate the performances of the training based on the extended Kalman filter in comparison with gradient-based learning.

KEYWORDS

Cramér-Rao bound, extended Kalman filter, feedforward neural networks, local minima, neural learning, performance metrics

1 | INTRODUCTION

Data-driven modeling aims at finding relationships among given data sets, and can be formulated with parametric or non-parametric machine learning algorithms. Parametric algorithms simplify the unknown function to be learned to a known form and contain a set of parameters of fixed size, which does not vary in dependence of the training examples. In particular, parametric algorithms do not require assumptions on the underlying probability distribution of the available data and thus can be employed independently of the data distribution. Such algorithms usually consist of two steps: selecting the form of the functional relationship, and learning the parameters from the training data. Examples are logistic regression, linear discriminant analysis, and naive Bayes (see, e.g., Chap. 2 of the book of James et al.[1] and Chap. 18 of the book of Russell and Norvig[2]). These methods are fast to learn from data as well as easy to understand and interpret. On the other hand, choosing *a priori* a functional form may limit their capability to match the underlying mapping generating the data, and therefore they are better suited for simple and low-dimensional problems. Instead, non-parametric machine learning models do not demand strong assumptions on the mapping generating the data and perform better when one has a lot of data and no prior knowledge, as they are able to fit a larger family of functional forms. However, they require much more training efforts in general, and it is often difficult to explain and interpret the obtained results. Examples are *k*-nearest neighbors regression, decision trees, deep Gaussian process, and complex neural networks, which are highly nonlinear.[1,2]

In this context, learning highly nonlinear models is usually based on some stochastic optimization method. Thus, the learned parameters are sensitive to their initial values, and can vary depending on different learning processes even with the same optimization algorithm. In this paper, we evaluate such an uncertainty in the data-driven learning process by using a metric based on an empirical measure of the capability to avoid local minima as well as Cramér-Rao bounds.[3] Thus, the goal of this manuscript is to develop a framework for data-driven analysis of effectiveness and robustness to local minima trapping of learning methods. Without loss of generality, we focus on one-hidden-layer feedforward neural networks, although our approach could be extended to other types of networks. To the best of our knowledge of the literature, this is the first attempt in this research direction. In more detail, the contribution of this paper is three-fold: (i) a novel, empirical criterion to evaluate robustness with respect to local minima trapping is proposed as a performance metric in the learning process of feedforward neural networks. Such an approach does not require assumptions on the underlying probability distribution function of the available data set; (ii) a learning algorithm based on the extended Kalman filter (EKF) is presented that is well-suited to avoiding local minima; (iii) the use of Cramér-Rao bounds is investigated to evaluate the convergence speed of neural learning.

For the purpose of learning, the knowledge of the function that generates the data distribution is of course helpful. However, a well-known limitation is that, if the chosen probability density function does not match the experimental dataset, bad predictive performances may occur. Instead, non-parametric approaches to density estimation with few assumptions about the distribution may turn out to be more robust. For example, the histogram is the simplest method for modelling probability distribution based on the observed data.[4,5] Standard histograms simply partition observed values into distinct bins of fixed width and then count the number of observations falling in each bin. In the present work, a novel criterion is proposed for the evaluation of the quality of neural training by computing the sample probability to avoid local minima.

As previously pointed out, we focus on the learning process of feedforward neural networks as a case study. Such a research direction is motivated by the slow convergence exhibited by classical backpropagation in high-dimensional settings.[6] In this context, a number of algorithms have been proposed in the literature for the purpose of training this kind of networks.[7] For example, the adaptive selection of the descent step[8] and the adoption of second-order derivatives methods were presented.[9] Among others, learning algorithms based on the EKF were also investigated.[10] The main advantage of EKF-based training is a higher convergence rate, which, however, is often obtained at the expense of a heavy computational effort (due to matrix inversions involved in the computation of EKF gains) and memory requirements. Thus, research efforts were devoted to face these drawbacks and improve flexibility of EKF-based learning in high-dimensional problems. As to computational issues, efficient EKF-learning techniques were developed for one-dimensional problems.[11] Multi-stream EKFs were addressed[10] to recursively train neural networks by using large amounts of data in batch mode (i.e., processing one data block at a time). The use of EKF training for recurrent neural networks was investigated as well.[12] Moreover, a connection was established between EKF and least squares for the purpose of neural training.[13] The solution to an optimal control formulation of online learning from supervised examples with regularization of the updates was also investigated and compared with the Kalman-filter estimate of the parameter vector to be learned.[14]

The choice of the EKF as learning algorithm is motivated by recent results,[15] where the EKF was used to select the optimal parameters of neural approximators with reduced computational effort and an increased robustness with respect to the trapping into local minima in an application involving the control of propagating interfaces modeled by level set methods.[16] Thus, this learning method is an interesting benchmark for the evaluation of the proposed metric aimed at assessing the capability to avoid the trapping into local minima of learning methods. In more detail, we regard neural network learning as a nonlinear state estimation problem, for which we investigate the performance of EKF-based training for the optimization of the neural parameters. Our study is focused on the robustness with respect to the local minima trapping and exploits the Cramér-Rao bound to investigate the effectiveness of the convergence speed to global minima. The approach is validated numerically on a problem involving the prediction of the Mackey-Glass series.[17]

The paper is organized as follows. In Section 2, we review one-hidden-layer feedforward neural networks with their related assumptions and approximation properties. Section 3 presents the considered EKF-based learning procedure as a nonlinear state estimation problem. Section 4 describes the proposed approach based on the Cramér-Rao bound to evaluate the performance of learning via the EKF, together with the definition of the sample probability metric to measure the capability to avoid local minima. Numerical results are discussed in Section 5. Conclusions are drawn in Section 6.

2 | FEEDFORWARD NEURAL NETWORKS

We consider the learning process of one-hidden-layer feedforward neural networks. Typically, computational units in the hidden layer of such networks are perceptrons, radial, and kernel units. One-hidden-layer networks are also know as shallow networks, in contrast to deep ones, in which there are two or more hidden layers. Although shallow networks present some limitations and deep ones are superior in certain tasks,[18] the one-hidden-layer paradigm is still widely used in applications.[16,19-21] The approximation capabilities of such networks were studied in several theoretical works. It was shown that, for many types of computational units, they are universal approximators, that is, they can approximate up to any desired accuracy every continuous function and every function belonging to Lebesgue spaces on compact subsets of $\mathbb{R}^p$, even for large p. For instance, this property holds for perceptrons with non-polynomial computational units,[22] as well as radial and kernel units satisfying mild conditions.[23] After the seminal paper of Barron,[24] the complexity issue, that is, the problem of estimating the number of computational units required to guarantee a desired accuracy in learning multidimensional mappings, was investigated too.[25]

A feedforward neural network with a single linear output computes input-output functions belonging to the set

$$\text{span } G := \left\{ \sum_{i=1}^{n} w_i g_i \ : \ w_i \in \mathbb{R}, \ g_i \in G \right\},$$

where G, called dictionary, is a parameterized family of functions, and $n \in \mathbb{N}$ is the number of units in the last hidden layer. In networks with one hidden layer (called "shallow"), G is formed by functions that are computable by a given type of computational units. In networks with several hidden layers (referred to as "deep"), instead, its elements are combinations and compositions of functions representing units from lower layers. Dictionaries are parameterized families of functions having the form

$$\{\varphi(\cdot, w) : U \to \mathbb{R} \mid w \in W\},$$

where $\varphi : U \times W \to \mathbb{R}$ is a function of two variables, i.e., an input vector $u \in U \subseteq \mathbb{R}^p$ and a parameter vector $w \in W \subseteq \mathbb{R}^k$. There are many possibilities to construct the parametrized functions $\varphi(\cdot, w)$ of the dictionary. Here, we consider the so-called ridge construction, in which the p-dimensional vector x is "shrunk" into a one-dimensional variable via the inner product, as follows:

$$\varphi(x, w_i) := h(x^\top \alpha_i + \beta_i),$$

where $w_i := \text{col}\,(\alpha_i, \beta_i) \in \mathbb{R}^{p+1}$. Thus, each ridge function results from the composition of a multivariable function having a particularly simple form given by the inner product $x^\top \alpha_i$, with a function h depending on one variable. The most widespread case is given by sigmoidal functions, that is, bounded measurable functions on the real line such that $\lim_{t\to+\infty} h(t) = 1$ and $\lim_{t\to-\infty} h(t) = 0$.

Let us denote by

$$\begin{aligned} \text{Ridge}(h) &:= \text{span } \left\{ h(x^\top \alpha_i + \beta_i), \alpha_i \in \mathbb{R}^p, \ \beta_i \in \mathbb{R} \right\} \\ &= \left\{ \gamma : \mathbb{R}^k \times \mathbb{R}^p \to \mathbb{R}; \ \ \gamma(w, x) = \sum_{i=1}^{n} c_i \ h(x^\top \alpha_i + \beta_i), \alpha_i \in \mathbb{R}^p, \ c_i, \beta_i \in \mathbb{R} \right\} \end{aligned} \tag{1}$$

the set of p-variable single-output ridge feedforward networks on $\mathbb{R}^p$ with a computational unit h, where $n \in \mathbb{N}$ is again the number of units (also called "neurons") in the hidden layer and $w \in \mathbb{R}^k$ collects all the parameters c_i, α_i, β_i. The next theorem is a slightly-simplified version of the results from Leshno et al.[26]

Theorem 1. *Let $h : \mathbb{R} \to \mathbb{R}$ be locally bounded and piecewise continuous and let n be a positive integer. Then, for every continuous function on a compact set $D \subset \mathbb{R}^p$ and every $\varepsilon > 0$, there exists a network in* Ridge(h), *that is, a ridge one-hidden-layer feedforward neural network (1) with a sufficiently large number n of computational units, that approximates such a function with an error at most ε in the supremum norm if and only if h is not an algebraic polynomial. The same holds for every function belonging to the Lebesgue spaces $\mathscr{L}_s(D)$, $s \in [1, \infty)$.*

In the neural network parlance, the property stated in Theorem 1 is the so-called "universal approximation" property, which is satisfied under mild hypothesis on the computational units of ridge networks. In particular, note that they do not need to be continuous or smooth, but the only requirement is non-polynomiality.

3 | EKF-BASED LEARNING

In this section, we present a framework based on the use of EKF for neural network learning. Although EKF-based learning has been successfully exploited in a variety of applications, its convergence properties are not fully known. In more detail, the exponential boundedness of the error in estimating the optimal network parameters was investigated.[27] Results on the convergence of certain recursive nonlinear least-squares algorithms in a purely deterministic context were reported, and the behavior of the EKF as a method for the identification of the optimal neural parameters was studied.[10]

Let us consider the problem of interpolating a data set given by M input-output pairs $\{(u_k, y_k),\ k = 0, 1, \dots, M-1\}$, where $u_k \in U \subset \mathbb{R}^p$ and $y_k \in Y \subset \mathbb{R}^q$ for two compact sets U and Y. We assume that each input-output pair is randomly generated via an unknown vector mapping $f : U \to Y$ such that $y_k = f(u_k)$. Suitable smoothness hypotheses on the input-output mapping f can be made according to the stochastic process generating the data.[28] For every component $1, \dots, q$ of f, a family $\Gamma_{\mathcal{N}}$ of parametrized mappings dependent on $\mathcal{N}(n)$ parameters, collected in the vector w, can be used to interpolate the data pairs, by searching for a convenient choice of the parameters.

Assumption 1. The family of parametrized mappings is given by $\Gamma_{\mathcal{N}(n)} = \text{Ridge}(h)$, where h is a computational unit that satisfies the hypotheses of Theorem 1 and $\mathcal{N}(n) = n(p+2)$.

Algorithm 1. EKF-based learning

Inputs: initial covariance matrix P_0;
neural network mapping $\gamma(\cdot, \cdot)$;
number of input/output pairs M;
input/output pairs (u_k, y_k), $k = 0, \dots, M-1$;
covariance matrix of the measurement noise R_k, $k = 0, \dots, M-1$.
Output: best estimate $\hat{w}_M$ of w^*.

Choose $\hat{w}_0$ randomly
for $k = 0$ **to** $M-1$ **do**
  $H_k \leftarrow \left.\frac{\partial \gamma(w,u)}{\partial w}\right|_{w=\hat{w}_k, u=u_k}$
  $K_k \leftarrow P_k H_k^\top \left(H_k P_k H_k^\top + R_k\right)^{-1}$
  $P_{k+1} \leftarrow P_k - K_k H_k P_k$
  $\hat{w}_{k+1} \leftarrow \hat{w}_k + K_k \left(y_k - \gamma\left(\hat{w}_k, u_k\right)\right)$
end for
return $\hat{w}_M$

Among the possible approaches to tune the parameter vector w, we focus on recursive least squares. Toward this end, we formulate an approximation process based on the definition of a dynamical system characterized by the following state and measurement equations:

$$w_{k+1} = w_k, \tag{2}$$

$$y_k = \gamma(w_k, u_k) + \eta_k, \tag{3}$$

where w_k is the neural parameter vector at iteration k and $k = 0, 1, \dots, M-1$. More specifically, the parameter values play the role of a constant state, with dynamic equation given by (2), equal to the optimal parameter vector $w^* \in \mathbb{R}^{\mathcal{N}(n)}$, that is, $w_0 = w_1 = \cdots = w^*$, while y_k represents the output variable described by (3), where $\eta_k \in \mathbb{R}^q$ is regarded as a random vector that models the difference between y_k and $\gamma(w^*, u_k)$, with γ being a neural mapping of the kind (1). Concerning

the definition of the optimal parameters, one can choose any w^* that is a minimizer of

$$\mathbb{R}^{\mathcal{N}(n)} \ni w \mapsto J(w) := \sum_{k=0}^{M-1} \|y_k - \gamma\ (w, u_k)\|^2 \in \mathbb{R}$$

over some compact subset of $\mathbb{R}^{\mathcal{N}(n)}$, where $J(\cdot)$ is a least-squares fitting cost.

Assumption 2. The mapping f generating the data pairs is locally bounded and piecewise continuous.

Note that Assumption 2 is quite reasonable: it is implied in the common situation in which one has no unbounded output values in the input-output pairs and the physical process generating them is discontinuous at most in correspondence of a finite number of input values.

Under Assumptions 1 and 2, Theorem 1 guarantees the existence of a ridge network and a parameter vector such that, for every $u \in U$ and every $\eta > 0$, the (unknown) mapping generating the data can be approximated with an error at most η in the supremum norm. The representation (2), (3) allows regarding supervised learning of neural networks on the basis of a set of data as the problem of estimating the state of a nonlinear system. The measurement equation defines the nonlinear relationship among inputs, outputs, and parameters.

Assumption 3. The computational unit h of the neural mapping Ridge(h) is continuously differentiable.

Algorithm 2. Computation of the mean Cramér-Rao Bound (MCRB)

Inputs: initial covariance matrix P_0;
neural network mapping $\gamma(\cdot, \cdot)$;
number of input/output pairs M;
input/output pairs (u_k, y_k), $k = 0, \dots, M-1$;
covariance matrix of the measurement noise R_k, $k = 0, \dots, M-1$;
number of simulation runs N.

Output: mean Cramér-Rao Bound MCRB_k, $k = 0, \dots, M-1$.

for $k = 0$ **to** $M-1$ **do**
  $\pi_k \leftarrow 0$
end for
for $i = 1$ **to** N **do**
  choose $\hat{w}_0$ randomly
  perform neural network training with initial weights $\hat{w}_0$ to get the final estimate w^*
  $\Pi_0 \leftarrow P_0$
  $\pi_0 \leftarrow \pi_0 + \text{tr}\,(\Pi_0)$
  for $k = 0$ **to** $M-1$ **do**
    $H_k^* \leftarrow \left.\frac{\partial \gamma(w,u)}{\partial w}\right|_{w=w^*, u=u_k}$
    $\Pi_{k+1} \leftarrow \Pi_k - \Pi_k H_k^{*\top} \left(H_k^* \Pi_k H_k^{*\top} + R_k\right)^{-1} H_k^* \Pi_k$
    $\pi_{k+1} \leftarrow \pi_{k+1} + \text{tr}\,(\Pi_{k+1})$
  end for
end for
for $k = 0$ **to** $M-1$ **do**
  $\text{MCRB}_k \leftarrow \pi_k / N$
end for
return MCRB_k, $k = 0, \dots, M-1$

Under Assumption 3, we introduce the EKF learning, as it is usually considered in the literature.[29] It consists in recursively computing estimates $\hat{w}_k \in \mathbb{R}^{\mathcal{N}(n)}$ of the optimal network parameters w^* for $k = 0, 1, \dots, M-1$, as follows:

$$\hat{w}_{k+1} = \hat{w}_k + K_k(y_k - \gamma\ (\hat{w}_k, u_k)), \tag{4}$$

$$H_k = \left.\frac{\partial \gamma(w,u)}{\partial w}\right|_{w=\hat{w}_k, u=u_k}, \tag{5}$$

$$K_k = P_k H_k^\top \left(H_k P_k H_k^\top + R_k\right)^{-1}, \tag{6}$$

$$P_{k+1} = P_k - K_k H_k P_k, \tag{7}$$

with symmetric positive definite matrices $P_k \in \mathbb{R}^{\mathcal{N}(n)\times\mathcal{N}(n)}$ and $R_k \in \mathbb{R}^{q\times q}$. The initialization is performed with a given initial parameter vector $\hat{w}_0$ and an initial covariance matrix P_0. The best estimate of w^* is obtained at the last iteration, that is, it is given by $\hat{w}_M$. The estimation procedure is sketched in Algorithm 1, which represents an implementation of Equations (4)–(7).

4 | PERFORMANCE EVALUATION

The performance of the EKF learning procedure described in Section 3 or of any training algorithm can be evaluated in terms of convergence speed and capability to avoid local minima trapping. In the following, we will address both issues.

As pretty well-known in the area of stochastic filtering, there exists a lower limit on the error covariance for unbiased estimators, called Cramér-Rao Bound (CRB). In our case, the covariance is given by

$$\mathrm{E}\left\{(\hat{w}_k - w^*)(\hat{w}_k - w^*)^\top\right\} \geq \Pi_k,$$

for $k = 0, 1, \ldots, M-1$, where $\mathrm{E}\{\cdot\}$ is the expectation operator and the positive definite matrix $\Pi_k \in \mathbb{R}^{\mathcal{N}(n)\times\mathcal{N}(n)}$ is computed as follows:[30]

$$\Pi_{k+1} = \Pi_k - \Pi_k H_k^{*\top}\left(H_k^* \Pi_k H_k^{*\top} + R_k\right)^{-1} H_k^* \Pi_k, \tag{8}$$

with

$$H_k^* := \left.\frac{\partial \gamma(w,u)}{\partial w}\right|_{w=w^*, u=u_k}$$

for $k = 0, 1, \ldots, M-1$ and $\Pi_0 = P_0$. The previous recursive equation coincides with Kalman formula, where the matrices H_k^* are computed in the optimal parameter vector w^*, and for this reason they are marked with a star.

The performance of EKF training (4)–(7) can then be evaluated by performing N simulations and computing the mean square error (MSE) at each iteration $k = 0, 1, \ldots, M-1$, that is,

$$\mathrm{MSE}_k(N) := \frac{1}{N}\sum_{i=1}^{N}(\hat{w}_k(i) - w^*(i))^\top(\hat{w}_k(i) - w^*(i)),$$

where, for the i-th simulation run, $w^*(i)$ and $\hat{w}_k(i)$ are the optimal parameters and the estimated ones at iteration k, respectively. The expected square error is the trace of Π_k, and therefore the following inequality holds for $k = 0, 1, \ldots, M-1$:

$$\lim_{N\to+\infty} \mathrm{MSE}_k(N) \geq \mathrm{tr}(\Pi_k).$$

The CRBs are lower bounds that are usually computed pathwise, but in our case the optimal parameters to which the EKF estimates asymptotically tend are unknown *a priori*. We can overcome such a difficulty by determining the CRB *a posteriori*, that is, in the i-th simulation run after finding $w^*(i)$ from some initial value. Moreover, we need to estimate the matrix R_k, as the variance of η_k is unknown (using for instance equation (36), p. 962, of the work of Iiguni et al.[31]). Since the CRB depends on R_k and also on the initial covariance matrix P_0, we focus on the mean CRB (denoted as MCRB in the following), that is, an average value obtained over many Monte Carlo simulation runs corresponding to different choices of the initial parameter vector. The pseudo-code of the procedure used for the computation of the MCRB is reported in Algorithm 2.

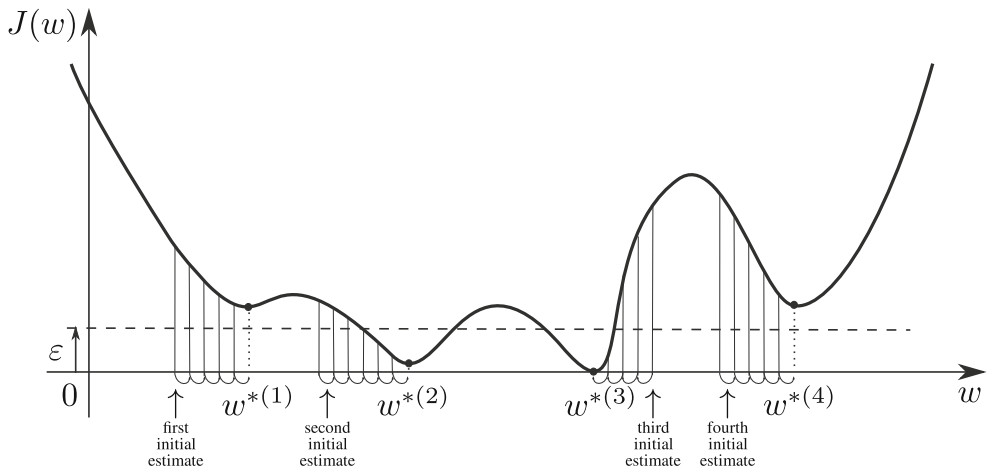

FIGURE 1 Sketch of a one-dimensional fitting cost function $J(w)$ with multiple local and global minima, denoted by $w^{*(1)}$, $w^{*(2)}$, $w^{*(3)}$, and $w^{*(4)}$. The value of ε that corresponds to the horizontal dashed line includes the minimum points $w^{*(2)}$ and $w^{*(3)}$. The application of a descent method from the second and third initial points leads to $w^{*(2)}$ and $w^{*(3)}$, respectively, and therefore such initial estimates have to be counted to compute the sample probability for the value of ε considered in the plot, while this is not the case of first and fourth ones.

In general, nonlinear regression problems suffer from local minima. This is typically the case when regression is performed by using neural networks. Thus, a learning algorithm may asymptotically provide different estimates of the optimal parameters, depending on both the particular realization of the stochastic processes that generate the data set and the choice of the initial parameter values. In this context, we propose a simple idea to evaluate the performance in minimizing the fitting cost J (ideally equal to zero since $\inf_{w\in\mathbb{R}^{\mathcal{N}(n)}} J(w) = 0$). It consists in evaluating the sample probability P_ε to find parameters for which the optimal cost is less than a given constant value ε. Thus, the sample probability P_ε provides a measure of the capability to avoid local minima. Figure 1 allows clarifying such an idea. In more detail, the probability of being trapped in local minima depends on the value of ε, which can be regarded as a threshold.

Given N initial estimates $\hat{w}_0(i)$, $i = 1, \dots, N$, let $\hat{w}_M(i)$ the corresponding final estimates obtained at iteration M. The sample probability P_ε we are interested in is defined as follows:

$$\mathbb{R}_{\geq 0} \ni \varepsilon \mapsto P_\varepsilon := \frac{1}{N}\sum_{i=1}^{N} \Theta(\varepsilon - J(\hat{w}_M(i))) \in [0, 1],$$

where $\Theta(\cdot)$ is the Heaviside function, that is, given $z \in \mathbb{R}$, $\Theta(z) = 1$ if $z \geq 0$, and $\Theta(z) = 0$ otherwise. Algorithm 3 reports the pseudo-code for the computation of P_ε.

5 | NUMERICAL RESULTS

For the purpose of numerical evaluation of the results described in previous sections, we considered the problem of prediction of the Mackey-Glass series,[17] which is a simple training case study yet meaningful as a first step toward the direction of data-driven performance criteria and metrics to evaluate both convergence speed and robustness to local minima trapping. The discrete-time Mackey-Glass series is generated by the following delay-difference equation

$$x_k = (1 - c_1)\, x_{k-1} + c_2\, \frac{x_{k-\tau}}{1 + (x_{k-\tau})^{10}}, \quad k = 0, 1, \dots, \tag{9}$$

where $\tau \geq 1$ is an integer constant, and $c_1 \in (0, 1)$, $c_2 > 0$ are given real constants. As is well-known, (9) shows a chaotic behavior, and its prediction is accepted as a standard benchmark problem. The prediction is performed by assuming that the next value x_{k+1} depends on a vector of constant length given by the previous l samples, that is, by interpolating the

Algorithm 3. Computation of P_ε

Inputs: threshold value ε;
neural network mapping $\gamma(\cdot,\cdot)$;
number of input/output pairs M;
input/output pairs $(u_k, y_k), k = 0, \dots, M-1$;
number of simulation runs N.
Output: sample probability P_ε to find parameters for which the fitting cost J is less than ε.

$P_\varepsilon \leftarrow 0$
for $i = 1$ **to** N **do**
  choose $\hat{w}_0$ randomly
  perform neural training with initial weights $\hat{w}_0$ to get the final estimate $\hat{w}_M$
  $J \leftarrow \sum_{k=0}^{M-1} \|y_k - \gamma(\hat{w}_M, u_k)\|^2$
  if $J < \varepsilon$ **then**
    $P_\varepsilon \leftarrow P_\varepsilon + 1/N$
  end if
end for
return P_ε

input-output data pairs $(u_k, y_k) \in \mathbb{R}^l \times \mathbb{R}$, where

$$u_k := \mathrm{col}(x_k, x_{k-1}, x_{k-2}, \dots, x_{k-l-1}) \mapsto y_k = x_{k+1}, \quad k = 0, 1, \dots .$$

In this paper, training data were generated with $\tau = 30$, $c_1 = 0.1$, and $c_2 = 0.2$. The initial values x_i, $i = -1, -2, \dots, -\tau$, were randomly chosen according to a uniform distribution between 0 and 0.4. We considered data sets composed of up to 5000 samples. The first 1000 pairs of each series were omitted. The succeeding data pairs were used half for training and half for testing. For the prediction we employed the previous five samples, that is, we chose $l = 5$. We used one-hidden-layer feedforwad neural networks with different numbers n of continuously differentiable sigmoidal computational units. Such networks were initialized with random choices of the initial parameters according to a Gaussian distribution with zero mean and covariance $P_0 = 10^{-2} I$, where I is the identity matrix.

Figure 2 reports the MCRBs obtained by the EKF learning over $N = 1000$ simulation runs and $M = 2000$ iterations, together with the behavior of the MSEs, for neural networks with different numbers n of neurons in the hidden layer. To enhance comparisons of results, we have reported normalized values, that is, MCRBs and MSEs are divided by the number of parameters. As expected, the larger the number n of neurons, the better the performance in terms of MSE, which exhibits a faster convergence rate as n increases.

We compared the results provided by the EKF learning with the standard gradient-based by-pattern adaptive training algorithm (the last one is available in Matlab via the function *traingda*). We denote such learning methods by EKFL (EKF learning) and GAL (gradient-based adaptive learning), respectively. We chose a number of iterations steps equal to 2000 for both EKFL and GAL, and the same initial parameters $\hat{w}_0$. The analysis of the behavior of EKFL and GAL by using the sample probability P_ε of being trapped into local minima is made via the plots reported in Figures 3, 4, and 5. In more detail, in Figure 3, we compared the P_ε profiles as functions of the threshold ε of EKFL and GAL obtained by neural networks with different numbers of computational units and a data set of $M = 1000$ input-output data pairs taken from the testing set. In Figure 4, we reported the sample probability P_ε provided by EKFL as a two-dimensional function depending on both the threshold ε and the number of neurons n for an easier comparison of the results. We observe that the best results in terms of P_ε are obtained with a neural network made up of $n = 10$ neurons. This is due to a compromise between approximation error and estimation error. A small value of n provides worse results, and this occurs also with neural networks having a higher complexity, that is, a larger n. In the first case, this may be ascribed to the poor approximation capability due to a reduced number of neurons.

In Figure 5, we evaluated the effect of the number M of data pairs on the sample probability P_ε, from $M = 200$ to $M = 2000$. It turns out that the approximation errors decrease as M increases. However, the precision of the learning deteriorates if neural networks with a large number n of neurons are considered due to the effect of local minima.

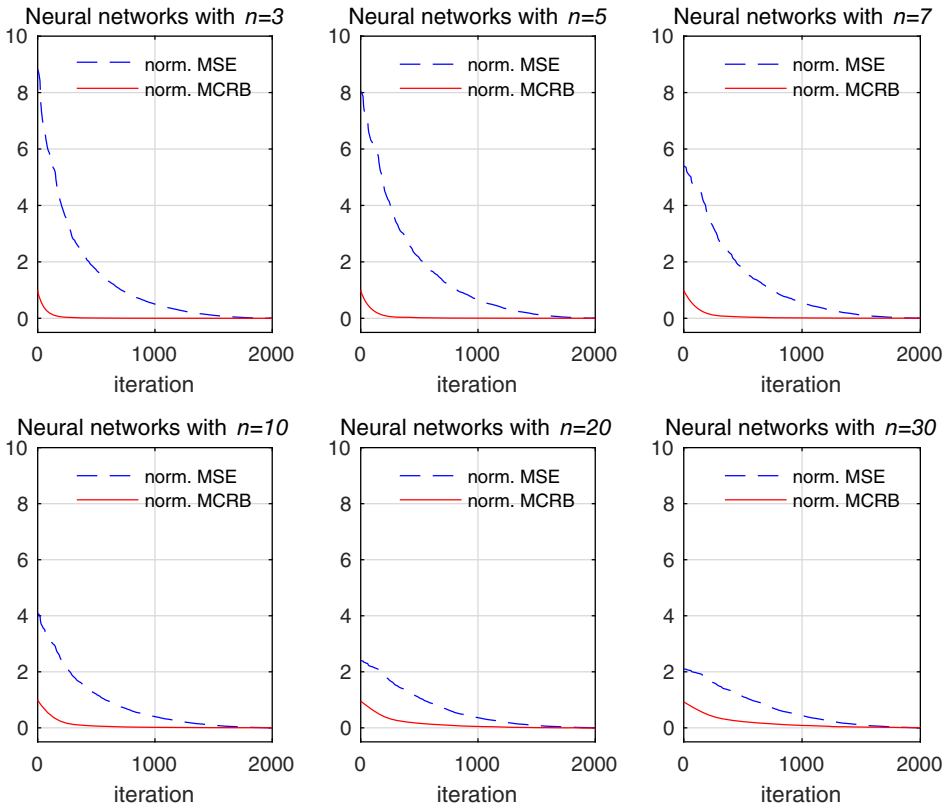

FIGURE 2 Normalized MSEs and MCRBs (MSEs and MCRBs divided by the number of weights to enhance comparisons) over $N = 1000$ simulation runs obtained by EKFL.

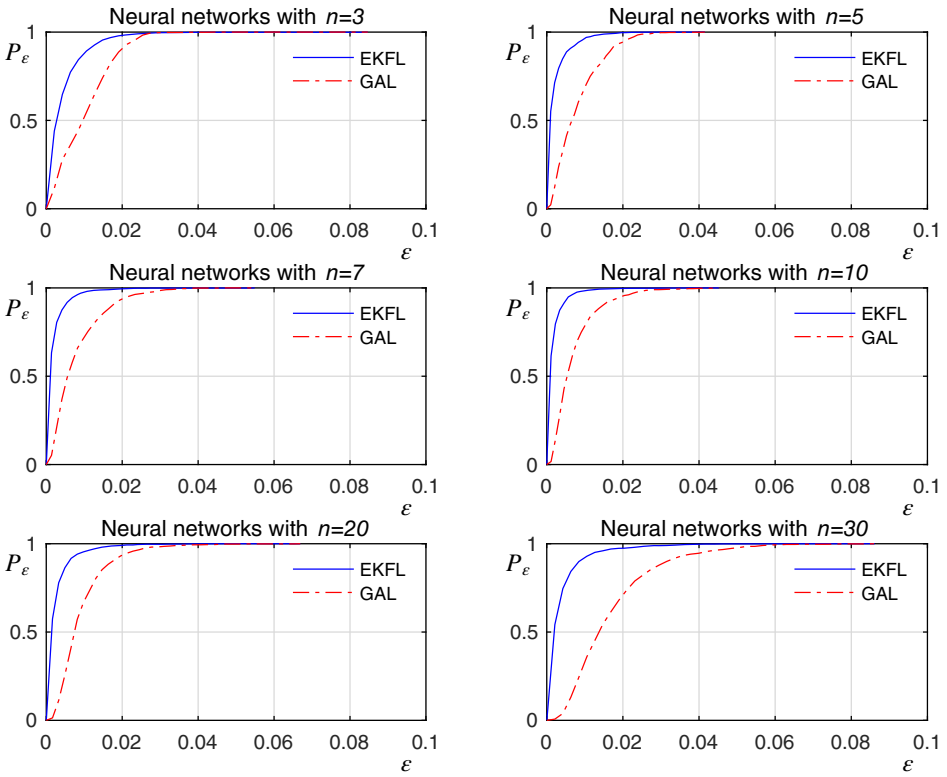

FIGURE 3 Sample probability P_ε of being trapped into local minima over $N = 1000$ simulation runs obtained by EKFL and GAL.

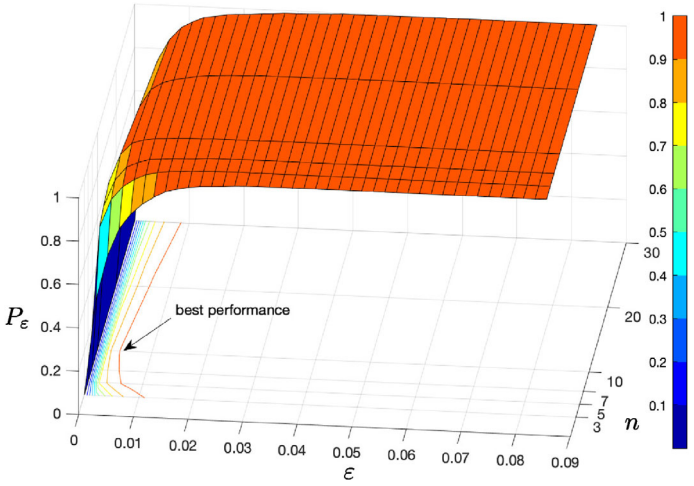

FIGURE 4 P_ε profiles as a two-dimensional function of the threshold ε and the number of neurons n obtained by EKFL over $N = 1000$ simulation runs.

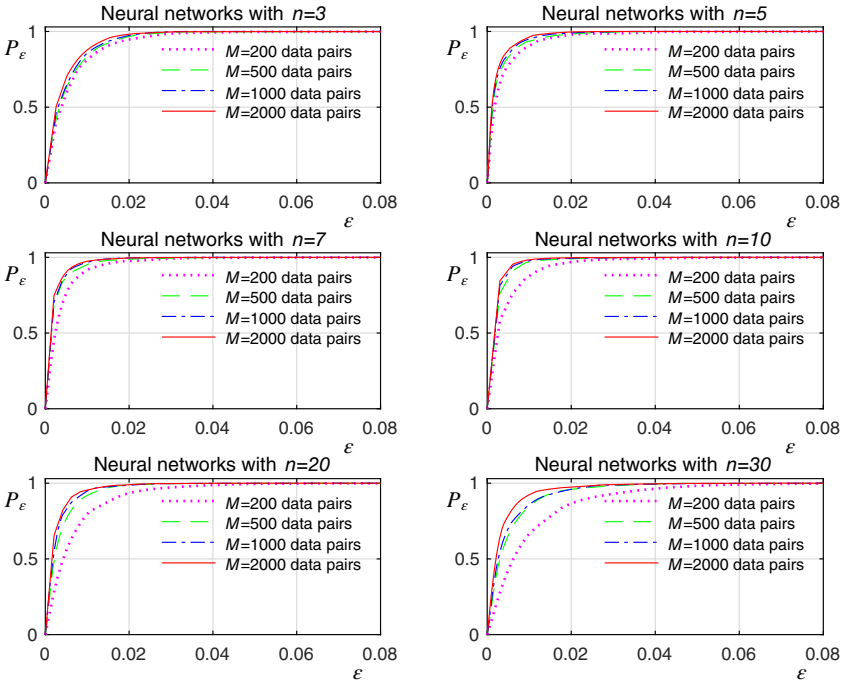

FIGURE 5 P_ε profiles as a function of the threshold ε over $N = 1000$ simulation runs obtained by EKFL with data sets of different sizes.

In fact, the corresponding values of P_ε are smaller as compared with those obtained with less complex neural networks. In general, it turns out that EKFL outperforms GAL in terms of capability to avoid local minima. Finally, the larger the number of samples, the better the results in terms of smaller local minima trapping probability. This is due to the positive effect of having at disposal a larger set of information when increasing the number of samples.

6 | CONCLUSIONS

Noises and uncertainties inevitably affect data-driven learning with possible deterioration of performance. In this paper, we have developed a framework for performance evaluation of neural learning, by focusing on one-hidden-layer feedforward neural networks and learning via the EKF as a case study. By exploiting an estimation-based point of view, we have adopted the Cramér-Rao bound, which provides the best convergence speed to be compared with. Another important point is the robustness with respect to local minima trapping, which has been measured in terms of the empirical probability of being attracted far from the global optimum. In this respect, as compared with gradient-based training, EKF learning performs quite well if a sufficiently large data set is available for training.

Summarizing, the main objective of this paper lies on the definition of data-driven procedures that can guide practitioners in understanding which is the best training algorithm in the specific problem of interest. In this context, future efforts will be devoted to the investigation of different training methods than the considered EKF learning and gradient-based adaptive learning, such as, for instance ADAM[32] or any other stochastic gradient descent algorithm. Another extension of the proposed approach will consist of investigating also local approximators based on kernel functions[33,34] and deep neural networks, by which we intend to deal with complex optimal control problems such as those addressed in previous works,[15,16] where training based on EKF was successfully applied, while taking into account the effect of accurate, smart sampling. Since EKF learning turns out to be well-suited to avoiding local minima, new training strategies based on EKF will be studied as well to overcome such local minima issues.

ACKNOWLEDGMENTS

This work is supported in part by the Italian Ministry of Enterprises and Made in Italy with project F/310027/01-03/X56, National Research Council of Italy with project PDGP DIT.AD021.104, and Italian Ministry of University and Research with projects ECS00000035 and PRIN 2022S8XSMY. Marcello Sanguineti is a member of GNAMPA-INdAM (Gruppo Nazionale per l'Analisi Matematica, la Probabilità e le loro Applicazioni–Istituto Nazionale di Alta Matematica) and Guest Scholar at IMT–School for Advances Studies (AXES Research Unit), Lucca.

ORCID

Angelo Alessandri https://orcid.org/0000-0001-6878-9106
Mauro Gaggero https://orcid.org/0000-0002-5048-4141
Marcello Sanguineti https://orcid.org/0000-0003-0355-8483

REFERENCES

1. James G, Witten D, Hastie T, Tibshirani R. *An Introduction to Statistical Learning*. Springer; 2017.
2. Russell S, Norvig P. *Artificial Intelligence: A Modern Approach*. Pearson; 2016.
3. Šimandl M, Královec J, Tichavský P. Filtering, predictive, and smoothing Cramér-Rao bounds for discrete-time nonlinear dynamic systems. *Automatica*. 2001;37(11):1703-1716.
4. Lista L. *Statistical Methods for Data Analysis: with Applications in Particle Physics*. Lecture Notes in Physics, Springer International Publishing; 2023.
5. Wu Y, Chen P. *Statistical Modeling in Biomedical Engineering*. Encyclopedia of Biomedical Engineering; Elsevier; 2019.
6. Fletcher R. *Practical Methods of Optimization*. Wiley; 1987.
7. Chow TWS, Cho SY. *Neural Networks and Computing: Learning Algorithms and Applications*. World Scientific; 2007.
8. Jacobs RA. Increased rates of convergence through learning rate adaption. *Neural Netw*. 1988;1:295-307.
9. Ampazis N, Perantonis S. Two highly efficient second-order algorithms for traning feedforward neural networks. *IEEE Trans Neural Netw*. 2002;13(5):1064-1074.
10. Alessandri A, Cuneo M, Pagnan S, Sanguineti M. A recursive algorithm for nonlinear least-squares problems. *Comput Optim Appl*. 2007;38(2):195-216.
11. De Nicolao G, Ferrari-Trecate G. Regularization networks: fast weight calculation via Kalman filtering. *IEEE Trans Neural Netw*. 2001;12(2):228-235.
12. Ilin R, Kozma R, Werbos P. Beyond feedforward models trained by backpropagation: a practical training tool for a more efficient universal approximator. *IEEE Trans Neural Netw*. 2008;19(6):929-937.
13. Xu Y, Wong K, Leung C. Generalized RLS approach to the training of neural networks. *IEEE Trans Neural Netw*. 2006;17(1):19-34.
14. Gnecco G, Bemporad A, Gori M, Sanguineti M. LQG online learning. *Neural Comput*. 2017;29:2203-2291.
15. Alessandri A, Bagnerini P, Gaggero M. Optimal control of propagating fronts by using level set methods and neural approximations. *IEEE Trans Neural Netw Learn Syst*. 2019;30(3):902-912.

16. Alessandri A, Bagnerini P, Gaggero M, Mantelli L, Santamaria V, Traverso A. Black-box modeling and optimal control of a two-phase flow using level set methods. *IEEE Trans Control Syst Technol*. 2022;30(2):520-534.
17. Mackey MC, Glass L. Oscillation and chaos in physiological control systems. *Science*. 1977;197:287-289.
18. Mhaskar H, Liao QL, Poggio T. *Learning Real and Boolean Functions: when Is Deep Better than Shallow*. Tech. rep., Center for Brains, Minds and Machines (CBMM) Memo No. 045. The Center for Brains, Minds and Machines; 2016.
19. Alessandri A, Cervellera C, Gaggero M. Predictive control of container flows in maritime intermodal terminals. *IEEE Trans Control Syst Technol*. 2013;21(4):1423-1431.
20. Orimoloye L, Sung MC, Ma T, Johnson J. Comparing the effectiveness of deep feedforward neural networks and shallow architectures for predicting stock price indices. *Expert Syst Appl*. 2020;139:112828.
21. Pineda-Jaramillo J. A shallow neural network approach for identifying the leading causes associated to pedestrian deaths in Medellin. *J Transp Health*. 2020;6:100912.
22. Pinkus A. Approximation theory of the MLP model in neural networks. *Acta Numer*. 1999;8:143-196.
23. Kůrková V, Kainen PC. Comparing fixed and variable-width Gaussian networks. *Neural Netw*. 2014;57:23-28.
24. Barron A. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Trans Inf Theory*. 1993;39(3):930-945.
25. Kainen PC, Kůrková V, Sanguineti M. Dependence of computational models on input dimension: tractability of approximation and optimization tasks. *IEEE Trans Inf Theory*. 2012;58:1203-1214.
26. Leshno M, Ya V, Pinkus A, Schocken S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Netw*. 1993;6(6):861-867.
27. Alessandri A, Cuneo M, Pagnan S, Sanguineti M. On the convergence of EKF-based parameters optimization for neural networks. Proc. IEEE Conf. On Decision and Control, Maui, Hawaii, USA. 2003 5825-5830.
28. Kurkova V, Sanguineti M. Approximate minimization of the regularized expected error over kernel models. *Math Oper Res*. 2008;33:747-756.
29. Leung CS, Tsoi AC, Chan LW. Two regularizers for recursive least squared algorithms in feedforward multilayered neural networks. *IEEE Trans Neural Netw*. 2001;12(6):1314-1332.
30. Kerr T. Status of CR-like lower bounds for nonlinear filtering. *IEEE Trans Aerosp Electron Syst*. 1989;25(5):590-601.
31. Iiguni Y, Sakai H, Tokumaru H. A real-time learning algorithm for a multilayered neural network based on the extended Kalman filter. *IEEE Trans Signal Process*. 1992;40(4):959-966.
32. Kingma D, Ba J. Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980. 2014.
33. Cervellera C, Gaggero M, Maccio D. Efficient kernel models for learning and approximate minimization problems. *Neurocomputing*. 2012;97:74-85.
34. Cervellera C, Gaggero M, Maccio D. Low-discrepancy sampling for approximate dynamic programming with local approximators. *Comput Oper Res*. 2014;43:108-115.

How to cite this article: Alessandri A, Gaggero M, Sanguineti M. Data-driven performance metrics for neural network learning. *Int J Adapt Control Signal Process*. 2023;1-12. doi: 10.1002/acs.3701