Certification of Autoencoder-based Models for Dynamical Systems

Marco Ledda, *Student, IEEE*, Diego Deplano, *Member, IEEE*,
Alessandro Giua, *Fellow, IEEE*, Mauro Franceschelli, *Senior, IEEE*

***Abstract*— Deep learning models have emerged as powerful tools for modeling complex dynamical systems, offering data-driven alternatives to traditional identification techniques. Among them, autoencoder-based architectures have gained popularity due to their ability to extract low-dimensional latent representations starting from high-dimensional information. However, a major challenge persists: assessing the reliability of these models, especially in control tasks where prediction errors can have critical consequences. In this work, we propose an optimization-based certification approach to quantify the worst-case prediction error of ReLU-activated autoencoder models of dynamical systems. By formulating a targeted Mixed-Integer Quadratic Programming, our approach identifies data sequences that maximize the deviation between the model's predicted output and the true system response.**

I. Introduction

Traditionally, dynamical systems have been modeled through analytical methods, deriving equations from physical laws to describe their behavior accurately. However, as systems grew in complexity-exhibiting nonlinear dynamics, high-dimensional states, or partially unknown behaviors, analytical methods became limited. This complexity, coupled with the availability of abundant data and computational advancements, motivated the adoption of data-driven system identification techniques, allowing to approximate system dynamics directly from input-output measurements [1–4]. These methods have been effective in a wide range of applications, particularly when the structure is known and its behavior can be captured easily.

The advent of machine learning (ML) and deep learning (DL) further accelerated this evolution. Deep neural networks (DNNs), due to their ability to perform function regression and approximate complex nonlinear mappings [5–9], have become a popular tool in this context, both in centralized and distributed scenarios [10, 11]. In particular, autoencoder-based architectures have been widely adopted because they are capable of learning compact latent representations [12]. In literature, a wide range of autoencoders have been proposed, from those that identify a nonlinear system in state-space [13–15] to variational autoencoders, whose resulting latent state is based on a probability distribution that has been learned from the data [16–18].

Despite their expressiveness, a key shortcoming of DL-based system models is the lack of formal guarantees. Unlike classical models, which often have well-established theoretical bounds on estimation or prediction error, deep learning models typically act as black boxes. This opacity raises significant concerns in critical applications such as autonomous systems, robotics, and processes. In particular, quantifying the *worst-case prediction error* is important for ensuring the model's trustworthiness and guaranteeing safety margins. To enhance safety guarantees in learned dynamical systems, various approaches have been explored. These include learning regions of attraction [19], exploiting Lyapunov theory for stability certification [20], and, in specific contexts, integrating physics-based knowledge into learning frameworks [21].

Main contribution – Despite the modeling accuracy of autoencoder-based architecture for dynamical systems, a critical limitation remains: the absence of guarantees on the worst-case one-step ahead prediction error. This is relevant in control applications, where wrong or inaccurate predictions can lead to unsafe or suboptimal decisions. Since this model type is used as surrogates in control applications such as Model Predictive Control [22–25], it's important to explore worst-case scenarios that lead to large prediction errors. The main contribution of this paper is to address this challenge by introducing a method to certify the maximum prediction error of autoencoder-based models for dynamical systems. We propose an optimization-based framework that identifies the sequences of data that maximize the discrepancy between the true system output trajectory and the one predicted by the ReLU-based autoencoder model. This method allows one to make a formal analysis of the worst-case behavior and provides a concrete error bound that can be used to evaluate the model's reliability. Our goal is to certify whether the predicted outputs from the autoencoder are consistent with those generated by the known physical system, under a given input-output trajectory window. The certification process identifies the worst-case deviation between the model and system outputs, providing guarantees on prediction fidelity.

Structure of the paper – The paper is organized as follows. In Section II we introduce the general problem and the mathematical formulation of the certification method, specifying the assumptions and the constraints involved. Section III shows the numerical simulations, starting from a method already proposed in the literature for the identification of Autoencoder-based models of dynamical systems. Section IV

The work of Diego Deplano was supported by the project e.INS-Ecosystem of Innovation for Next Generation Sardinia (cod. ECS 00000038) funded by the Italian Ministry for Research and Education (MUR) under the National Recovery and Resilience Plan (NRRP) - MISSION 4 COMPONENT 2, "From research to business" INVESTMENT 1.5, "Creation and strengthening of Ecosystems of innovation" and construction of "Territorial R& D Leaders".

All authors are with the DIEE, University of Cagliari, 09123 Cagliari, Italy. Emails: marco.ledda2@unica.it, diego.deplano@unica.it, giua@unica.it, mauro.franceschelli@unica.it. Mauro Franceschelli is the corresponding author.

© 2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. DOI: 10.1109/CDC57313.2025.11312089

draws the conclusion and outlines future research directions.

Notation – The set of real and natural numbers is denoted by $\mathbb{R}$ and $\mathbb{N}$. Matrices $M \in \mathbb{R}^{n\times n}$ are denoted by uppercase letters, vectors $v \in \mathbb{R}^n$ by boldface bold letters, scalars $s \in \mathbb{R}$ by lowercase letters, while sets and spaces $\mathcal{S}$ are denoted by uppercase calligraphic letters. The vectors of zeros and ones of dimension n are denoted by $\mathbb{0}$ and $\mathbb{1}$, respectively. The Euclidean norm is denoted by $\|\cdot\|_2$. The operators $\max$ and $\min$ refer to the maximum and minimum of a finite set of real numbers. The operators $\sup$ and $\inf$ denote the least upper bound (supremum) and greatest lower bound (infimum), respectively, and are used when the extrema may not be attained but can be approached arbitrarily closely. The symbol $\circ$ denotes function composition: for two functions f and g, the composition $f \circ g$ is defined as $(f \circ g)(x) = f(g(x))$.

II. Problem statement and proposed method

We consider dynamical systems with state space representation

$$\begin{cases} x_{k+1} = f(x_k, u_k) \\ y_k = g(x_k) \end{cases} \tag{1}$$

where $x_k \in \mathbb{R}^{n_x}$ is the state vector, $u_k \in \mathbb{R}^{n_u}$ is the exogenous input, and $y_k \in \mathbb{R}^{n_y}$ is the output, with $n_x, n_u, n_y \in \mathbb{N}$. We also consider artificial neural networks (ANN) that could make predictions $\hat{y}_k$ of the system's outputs y_k based on the knowledge of a sequence of input/output pairs $\{y_{k-T}, u_{k-T}, \cdots, y_{k-1}, u_{k-1}\}$ of length $T \in \mathbb{N}$. Namely, denoting by $f_{\mathsf{ann}} : \mathbb{R}^{T(n_u+n_y)} \to \mathbb{R}^{n_y}$ the mapping of the ANN we have

$$\hat{y}_k = f_{\mathsf{ann}}(y_{k-T}, u_{k-T}, \cdots, y_{k-1}, u_{k-1}) \tag{2}$$

In general, the ANN will commit an error in estimating the future output of the dynamical system, and we denote this error by

$$\varepsilon_k = \|\hat{y}_k - y_k\|_2^2. \tag{3}$$

We can now formalize the main problem of interest.

Problem 1 (Certification problem). *What is the largest prediction error* (3) *that a given ANN* (2) *can make when predicting the one-step ahead output of a dynamical system* (1)*?*

To solve the above certification problem, one needs to find the input/output sequence that leads to the largest prediction error between the true system's output y_k and the ANN's estimated output $\hat{y}_k$. Letting $k = T$ be the time at which we seek to make the prediction, the past inputs $\mathcal{U}_p = \{u_0, \ldots, u_{T-1}\}$ becomes free decision variables while the system's past output $\mathcal{Y}_p = \{y_0, \ldots, y_{T-1}\}$ becomes constrained decision variables determined by the known model dynamics (1). The associated state trajectory is denoted by $\mathcal{X}_T = \{x_0, \ldots, x_T\}$. In this way, both the true output y_T and the ANN predicted output $\hat{y}_T$ become functions of the so-called *information vector* $[y_0^\top, u_0^\top, \cdots, y_{T-1}^\top, u_{T-1}^\top]^\top$ [13]. With this setting, in principle, an answer to Problem 1 can be found by solving the following optimization problem:

$$\max_{\mathcal{U}_p, \mathcal{Y}_p, \mathcal{X}_T, y_T, \hat{y}_T} \quad \|\hat{y}_T - y_T\|_2^2 \tag{4a}$$

$$\text{s.t.} \quad x_0 = \mathbb{0}, \tag{4b}$$

$$\forall k \in \{0, \cdots, T-1\}: \quad y_k = g(x_k),\ x_{k+1} = f(x_k, u_k), \tag{4c}$$

$$y_T = g(x_T) \tag{4d}$$

$$\hat{y}_T = f_{\mathsf{ann}}(y_0, u_0, \cdots, y_{T-1}, u_{T-1}) \tag{4e}$$

The above problem allows us to find the sequence of inputs such that, starting from x_0, we reach an unknown state x_T where the distance between the output prediction at time T, $\hat{y}_T$, and the real output y_T is maximized. However, it could be very difficult to solve exactly the above optimization problem, due to possible nonlinearities of both the system's dynamics and the ANN architecture.

In the remainder of this section, we will show that Problem 1 can be formulated as a Mixed-Integer Quadratic Programming (MIQP) for piecewise linear systems and feedforward ANNs with Leaky Rectified Linear Unit (LReLU) activation functions, as formalized next.

Assumption 1. *The dynamical system* (1) *has a piecewise linear dynamics and a bounded state space $\mathcal{X} \subset \mathbb{R}^{n_x}$ and bounded input space $\mathcal{U} \subset \mathbb{R}^{n_u}$. Let $\mathcal{X} = \mathcal{X}_1 \cup \cdots \cup \mathcal{X}_m$ be a partition of the state space into convex polyhedra $\mathcal{X}_i = \{x \in \mathbb{R}^n : \Pi_i x \le \pi_i\}$, then the dynamics can be written as follows*

$$x_{k+1} = \begin{cases} A_1 x_k + B_1 u_k & \text{if } x_k \in \mathcal{X}_1, \\ \vdots & \\ A_m x_k + B_m u_k & \text{if } x_k \in \mathcal{X}_m \end{cases}$$

and output $y_k = C x_k$.

Assumption 2. *The ANN* (2) *is a fully-connected feedforward neural network made up of $L \in \mathbb{N}$ layers with $n_\ell \in \mathbb{N}$ neurons for $\ell \in [1, L]$ such that*

$$f_{\mathsf{ann}} := f_{L+1} \circ \cdots \circ f_\ell \circ \cdots \circ f_1,$$

where the functions $f_\ell : \mathbb{R}^{n_{\ell-1}} \to \mathbb{R}^{n_\ell}$ are LReLU, with $n_0 = T(n_u + n_y)$ and $n_{L+1} = n_y$, i.e.,

$$f_\ell(h) = \mathrm{LReLU}_\alpha(W^\ell h + b^\ell), \quad h \in \mathbb{R}^{\ell-1},$$

and where

$$\mathrm{LReLU}_\alpha(x) := \max\{\alpha x, x\}, \quad \textit{and} \quad \alpha \in [0, 1].$$

A. Formulation of the MIQP problem

We start by discussing how the constraints (4c) associated with the decision variables x_k, which are described by a piecewise affine model as in Assumption 2, can be written as linear constraints by following the approach outlined in [26]. For each partition $\mathcal{X}_j = \{x \in \mathbb{R}^n : \Pi_j x \le \pi_j\}$ and for each step $k = 0, \cdots, T-1$ and each partition $j = 1, \cdots, m$ let $\delta_{k,j} \in \{0, 1\}$ be an auxiliary boolean variable that is equal to 1 if and only if $x_k \in \mathcal{X}_j$, namely

$$\delta_{k,j} = 1 \quad \Rightarrow \quad \Pi_j x_k - \pi_j \le \mathbb{0}.$$

© 2025 IEEE. Personal use of this material is permitted. DOI: 10.1109/CDC57313.2025.11312089

This can be equivalently expressed by the mixed-integer linear inequalities:

$$\forall k, j: \qquad \Pi_j x_k - \pi_j \leq U'(1-\delta_{k,j})\mathbb{1} \tag{5}$$

where

$$U' := \max_{j=1,\cdots,m} \left\{ \max \left\{ \sup_{x\in\mathcal{X}} \{\Pi_j x - \pi_j\} \right\} \right\}.$$

Since all partitions $\mathcal{X}_i$ are disjoint by definition, it must hold

$$\forall k: \quad \sum_{j=1}^{m} \delta_{k,j} = 1. \tag{6}$$

Now, let $z_{k,j} \in \mathbb{R}^{n_x}$ be continuous variables such that

$$z_{k,j} = \begin{cases} A_j x_k + B_j u_k & \text{if } \delta_{k,j} = 1, \\ \mathbb{0} & \text{if } \delta_{k,j} = 0. \end{cases}$$

This can be equivalently expressed by four mixed-integer linear inequalities:

$$\forall k, j: \quad \begin{cases} z_{k,j} \leq U''\delta_{k,j}\mathbb{1}, \\ z_{k,j} \geq L''\delta_{k,j}\mathbb{1}, \\ z_{k,j} \leq A_j x_k + B_j u_k - L''(1-\delta_{k,j})\mathbb{1}, \\ z_{k,j} \geq A_j x_k + B_j u_k - U''(1-\delta_{k,j})\mathbb{1}, \end{cases} \tag{7}$$

where

$$U'' := \max_{j=1,\cdots,m} \left\{ \max \left\{ \sup_{x\in\mathcal{X}, u\in\mathcal{U}} \{A_j x - B_j u\} \right\} \right\},$$
$$L'' := \min_{j=1,\cdots,m} \left\{ \min \left\{ \inf_{x\in\mathcal{X}, u\in\mathcal{U}} \{A_j x - B_j u\} \right\} \right\}.$$

Thus, the state x_{k+1} can be expressed as a function of the $z_{k,j}$ as follows

$$\forall k: \quad x_{k+1} = \sum_{j=1}^{m} z_{k,j}. \tag{8}$$

We summarize these equivalencies in the following proposition.

Proposition 1. *Consider the optimization problem* (4) *under Assumption 1. All constraints on the output variables in* (4b)-(4d) *becomes linear:*

$$y_0 = \mathbb{0}, \quad y_k = Cx_k, \quad \forall k = 1, \cdots, T-1. \tag{9}$$

By introducing boolean variables $\delta_{k,j} \in \{0,1\}$ and continuous variables $z_{k,j} \in \mathbb{R}^{n_x}$ for all $k = 1, \cdots, T-1$ and $j = 1, \cdots, m$, all the constraints on the state variables in (4c) *can be replaced by the linear constraints in eqs.* (5)-(8).

We continue by discussing how the constraint (4e) associated with the prediction variable $\hat{y}_k$, which is the result of LReLU functions as in Assumption 2, can be decomposed into a set of linear constraints due by following the approach outlined in [27]. We first associate a continuous variable $h^\ell \in \mathbb{R}^{n_\ell}$ to the output of each layer $\ell \in 1, \cdots, L$, namely

$$h^\ell := \max\{\alpha(W^\ell h^{\ell-1} + b^\ell), (W^\ell h + b^\ell)\}, \quad \text{and} \quad \alpha \in [0,1],$$

where the input to the first layer is the information vector and the output of the last layer is the expected output, namely

$$\begin{aligned} h^0 &= [y_0^\top, u_0^\top, \cdots, y_{T-1}^\top, u_{T-1}^\top]^\top, \\ \hat{y}_T &= f_{L+1}(h^L). \end{aligned} \tag{10}$$

The max operation can be replaced by introducing continuous variables $z_\ell \in \mathbb{R}^{n_\ell}$ such that

$$\forall \ell: \quad z^\ell = W^\ell h^{\ell-1} + b^\ell \tag{11}$$

By construction, the maximum $c = \max\{a, b\}$ of two terms is always greater or equal than all arguments, namelt $c \geq a$ and $c \geq b$, and therefore one must impose

$$\forall \ell: \quad h^\ell \geq \alpha z^\ell, \quad \text{and} \quad h^\ell \geq z^\ell. \tag{12}$$

Finally, by introducing boolean variables $\delta_\ell \in \{0,1\}^{n_\ell}$ one may force each component h_i^ℓ to be equal to z_i^ℓ when $\delta_{\ell,i} = 1$ and equal to αz_i^ℓ when $\delta_{\ell,i} = 0$ with the following set of mixed-integer linear constraints

$$\forall \ell: \quad h^\ell \leq \alpha z^\ell + M\delta_\ell, \quad \text{and} \quad h^\ell \leq z^\ell + M(1-\delta_\ell), \tag{13}$$

where

$$M := \max_{\ell=1,\cdots,L} \left\{ \max \left\{ \sup_{x\in\mathcal{X}} \left\{ \alpha(W^\ell x + b^\ell), W^\ell x + b^\ell \right\} \right\} \right\}.$$

We summarize these equivalences in the following proposition.

Proposition 2. *Consider the optimization problem* (4) *under Assumption 2. By introducing boolean random variables $\delta_\ell \in \{0,1\}^{n_\ell}$ and continuous variables $h^0 \in \mathbb{R}^{T(n_u+n_y)}$, $h^\ell, z^\ell \in \mathbb{R}^{n_\ell}$ for all $\ell = 1, \cdots, L$, the constraint on the prediction variable in* (4e) *can be replaced by the the linear constraints in eqs.* (10)- (13).

Having expressed both the system dynamics and the ANN prediction through mixed-integer linear constraints, the certification problem is defined as a MIQP. Before solving this problem in simulation, we must ensure that it is always feasible under the given assumptions. This is formalized in the next proposition, while a rigorous proof will be provided in future work.

Proposition 3. *Consider the optimization problem* (4) *under Assumption 2. For any bounded input and state domains $\mathcal{U}$ and $\mathcal{X}$, the certification problem* (4) *is always feasible. In particular, the system dynamics constraints* (4b)–(4d) *are satisfied by construction, while the ANN predictor* (4e) *is always defined for any admissible input-output sequence.*

B. Discussion on the implementation and complexity

Several aspects influence the practical implementation of the proposed certification method. First, the formulation requires the determination of the bounds L'', U', U'', and M, which appear in the linear constraints related to the piecewise affine dynamics and the LReLU activation functions. These bounds can be computed starting from the prior knowledge of the training set. In practice, since the dataset defines the region of the state space $\mathcal{X}$ and the input space $\mathcal{U}$ used

© 2025 IEEE. Personal use of this material is permitted. DOI: 10.1109/CDC57313.2025.11312089

during the training stage, one can pre-compute these values, ensuring that they enclose the operating ranges observed in the data [28, 29]. Another important aspect is the low computational complexity of the resulting MIQP problem. The number of variables and the scale are linearly with:

- The length T of the input/output sequence;
- The dimension n_u of the dynamical system's input;
- The dimension n_y of the dynamical system's output;
- The dimension n_x of the dynamical system's state;
- The number m of the dynamical system's linearity regions;
- The total number $n_L = \sum_{\ell=1}^{L} n_\ell$ of neurons in all layers.

In a detailed count, the number of boolean variables scales with $\mathcal{O}(T \cdot m + n_\ell)$, while both the number of real variables and the number of constraints scale with $\mathcal{O}(T(m \cdot n_x + n_y + n_u) + n_\ell)$. Furthermore, while our formulation employs the squared Euclidean norm in the objective function to quantify the prediction error, alternative norms such as the ℓ_1 or ℓ_∞ norms could be used instead to make the objective convex.

III. Numerical simulations with Autoencoders

This section presents numerical simulations of the proposed certification method in the case the ANN is modeled as an autoencoder satisfying Assumption 2. An autoencoder is a specific kind of ANN that consists of two cascaded ANNs (see Figure 1):

- The encoder $e : \mathbb{R}^{T(n_y+n_u)} \to \mathbb{R}^{n_{\Lambda+1}}$, which maps a sequence of input/output past pairs $[u, y]_{k-T}^{k-1}$ into a latent representation $\hat{x}_k \in \mathbb{R}^{n_{L+1}}$ of the true system's state $x_k \in \mathbb{R}^{n_x}$ with $n_{L+1} \geq n_x$ (and usually $n_{\Lambda+1} \gg n_x$);
- The decoder $d : \mathbb{R}^{n_{L+1}} \to \mathbb{R}^{n_y}$, which makes a prediction $\hat{y}_k \in \mathbb{R}^{n_y}$ of the output $y_k \in \mathbb{R}^{n_y}$ from the latent representation $\hat{x}_k$ of the system's state.

More precisely, both the encoder and the decoder are designed with the same structure as follows:

- A number of layers equal to $\Lambda = 3$, each with ReLU activation functions, i.e., LReLU with $\alpha = 0$, followed by a final linear output layer (with this notation, the ANN has a total number of layers equal to $L = 2\Lambda+1$);
- A number of neurons equal to $n_\ell = 6$ for all layers $\ell = 1, \ldots, \Lambda$;
- A dimension of the latent state space equal to $n_{\Lambda+1} = 2n_x$ where n_x is the dimension of the model's state;
- An observation window of length $T = 100$, i.e., inputs and outputs are observed at times $k = 0, 1, \cdots, T-1$.

Throughout this section, we train[1] autoencoders by using the technique outlined in [13]. We constructed training and validation datasets of 20.000 and 2.000 samples, respectively, by exciting the system with sequences of step signals of length $T = 100$ with random amplitudes drawn from the Gaussian distribution with zero mean and variance equal to 1. Following data generation, all signals were subjected to a standard normalization procedure to ensure consistency. In particular, the empirical mean and standard deviation were computed from the training dataset and used to normalize both training and validation sets by subtracting the mean and dividing by the standard deviation. We also added Gaussian white noise with zero mean and a standard deviation of $\sigma = 0.02$ to both the training and validation datasets.

[1]The software is available at `http://dysco.imtlucca.it/masti/autoencoders`

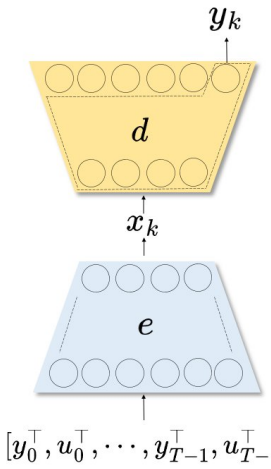


Fig. 1: Schematic representation of the autoencoder's architecture [13]. The encoder block is fed with the information available up to time step $k-1$. The decoder block reproduces the observable output starting from the latent state x_k.

To evaluate the proposed method of certification, we test it on the following kind of dynamical systems:

1) A linear system described in Section III-A;
2) A piecewise linear system described in Section III-B;
3) A nonlinear system described in Section III-C.

We discuss the results in Section III-D.

A. Example 1: A linear dynamical systems

The discrete-time linear time-invariant system is described by

$$\Sigma_{LTI} := \begin{cases} x_{k+1} = \begin{bmatrix} 0.95 & -0.1 & 0 \\ 0.1 & 0.95 & 0 \\ 0 & 0 & 0.9 \end{bmatrix} x_k + \begin{bmatrix} 1 \\ 0.5 \\ 0 \end{bmatrix} u_k \\ y_k = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x_k \end{cases} \tag{14}$$

The system was tested with inputs $u_k \in [0, 2] := \mathcal{U}$ which resulted in a tested state space included within $\mathcal{X} = [-15 \cdot \mathbb{1}, +15 \cdot \mathbb{1}]$. The result of the certification method are shown in Fig. 2, revealing that the worst quadratic error the ANN could make in predicting the system's output is 56.40.

© 2025 IEEE. Personal use of this material is permitted. DOI: 10.1109/CDC57313.2025.11312089

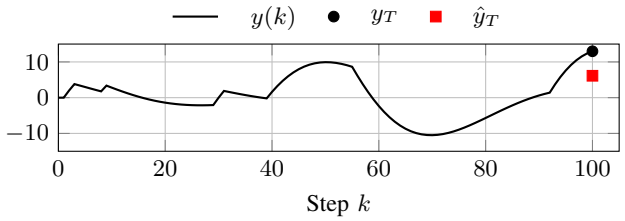


Fig. 2: Numerical simulation for system Σ_{LTI} in eq. (14). The curve represents the observation until time step $k = 99$. The discrepancy is observed between time steps $k = 99$ and $k = 100$. The two points represent the discrepancy between the systems.

B. Example 2: A piecewise affine linear system

The piecewise affine linear system is a gear system [26]. Let:

- $x_k \in \mathbb{R}$ denote the vehicle speed at discrete time step k;
- $q_k \in \{1, 2, 3\}$ denote the current gear;
- $u_k \in \mathbb{R}$ denote the control input, representing the accelerator effort.

The evolution of the speed vehicle is given by the state equation:

$$\Sigma_{PWA} := \begin{cases} x_{k+1} = a_{q_k} x_k + b u_k - d \\ y_k = x_k \end{cases} \tag{15}$$

where a_{q_k} is a gear-dependent coefficient representing the gain associated with the current gear, $b \in \mathbb{R}_{\geq 0}$ represents the effect of the acceleration input, and $d \in \mathbb{R}_{\geq 0}$ is a constant deceleration factor modeling friction. The transition of the gear state q_k is governed by the following logic based on the vehicle's speed:

$$\begin{cases} \text{If } q_k = 1 \text{ and } x_{k+1} > \theta_1, & q_{k+1} = 2, \\ \text{If } q_k = 2 \text{ and } x_{k+1} > \theta_2, & q_{k+1} = 3 \end{cases}$$

where θ_1, θ_2 are the gear-up speed thresholds. Vice versa, when the gear shifts down, the logic is the following:

$$\begin{cases} \text{If } q_k = 3 \text{ and } x_{k+1} < \phi_1, & q_{k+1} = 2, \\ \text{If } q_k = 2 \text{ and } x_{k+1} < \phi_2, & q_{k+1} = 1 \end{cases}$$

where ϕ_1, ϕ_2 are the gear-down speed thresholds. The system was tested with inputs $u_k \in [-4, 4] := \mathcal{U}$ which resulted in a tested state space included within $\mathcal{X} = [\mathbb{0}, 10 \cdot \mathbb{1}]$. The result of the certification method are shown in Fig. 3, revealing that the worst quadratic error the ANN could make in predicting the system's output is 58.98.

C. Example 3: A nonlinear system

The discrete-time nonlinear system consists of two water tanks [30] described by the following equations:

$$\Sigma_T := \begin{cases} x_{k+1,1} = x_{k,1} - k_1 \sqrt{x_{k,1}} + k_2 u_k \\ x_{k+1,2} = x_{k,2} + k_3 \sqrt{x_{k,1}} - k_4 \sqrt{x_{k,2}} \\ y_k = x_{k,2} \end{cases} \tag{16}$$

The cascaded tank system is a fluid-level control system comprising two tanks with free outlets by a pump. The input signal controls the pump that delivers the water to the first tank. The water from the first tank then flows down to the lower tank through a small opening. This system does not consider possible overflows. We set $k_1 = 0.5$, $k_2 = 0.4$, $k_3 = 0.2$, $k_4 = 0.3$. The system was tested with inputs $u_k \in [-4, 4] := \mathcal{U}$ which resulted in a tested state space included within $\mathcal{X} = [\mathbb{0}, 10 \cdot \mathbb{1}]$. The result of the certification method are shown in Fig. 4, revealing that the worst quadratic error the ANN could make in predicting the system's output is 31.92.

D. Discussion of the results

The numerical simulation results provide several insights into the performance and limitations of the autoencoder-based prediction scheme. In all three cases – the linear system, the piecewise linear system, and the nonlinear system – the observed data span the interval from $k = 0$ to $k = 99$, with the prediction occurring at $k = 100$. The figures (2) (3) (4), illustrate a clear discrepancy at this prediction point: the red marker, corresponding to the autoencoder's prediction, consistently deviates from the black marker that indicates the true system output. This difference is quantitatively measured by the squared error, which is the value of the objective function in our optimization problem. In the nonlinear case, the nonconvex nature of the problem brings us to say that we have found a lower bound for the worst-case scenario, but this does not necessarily guarantee to have found a global optimal solution. Lastly, it is important to note that the certification method, implemented using the Gurobi [31] solver, quantifies these errors by optimizing the sequence of past inputs and outputs. The obtained objective function value not only serves as a bound on the prediction error but also provides actionable insights into the reliability of the autoencoder in control applications. For a detailed summary of the solving time and objective values obtained in our simulations, please refer to Table I. Note that the experiments[2] have been run with a workstation equipped with an Intel Core i9 and 32 GB of RAM.

[2]Python code of the implementation is available at: `https://github.com/mledda17/AE-Certification-Method`.

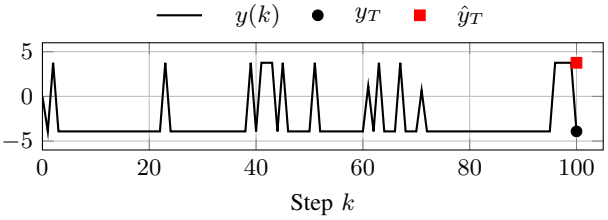


Fig. 3: Numerical simulation for system Σ_{PWA} in eq. (15). The curve represents the observation until time step $k = 99$. The discrepancy is observed between time steps $k = 99$ and $k = 100$. The two points represent the discrepancy between the systems.

© 2025 IEEE. Personal use of this material is permitted. DOI: 10.1109/CDC57313.2025.11312089

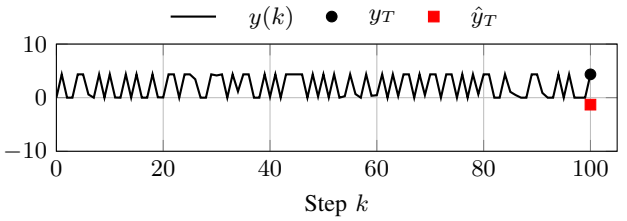


Fig. 4: Numerical simulation for system Σ_T in eq. (16). The curve represents the observation until time step $k = 99$. The discrepancy is observed between time steps $k = 99$ and $k = 100$. The two points represent the discrepancy between the systems.

Objective function values and solving time		
System	**Objective value**	**Solving time**
Example 1	56.40	0.26 s
Example 2	58.98	0.90 s
Example 3	31.92	4.14 s

TABLE I: In this table are reported, for each system, the value of the objective function when a solution has been found and the corresponding solving time.

IV. Conclusions

In conclusion, we presented an optimization-based certification method that quantifies the worst-case one-step ahead prediction error of autoencoder-based models for dynamical systems. By formulating and solving the problem, our approach provides a formal bound on the discrepancy between the autoencoder predictions and the true system output. The discrepancy highlighted in the results emphasizes the need for worst-case analysis for this type of application, especially when applied in critical fields like control. These findings underscore the trade-offs related to using deep learning models for system identification and highlight future research direction.

References

[1] K. Åström and P. Eykhoff, "System identification—a survey," *Automatica*, vol. 7, no. 2, pp. 123–162, 1971.

[2] J. Schoukens and L. Ljung, "Nonlinear system identification: A user-oriented roadmap," *CoRR*, vol. abs/1902.00683, 2019.

[3] D. Piga, M. Mejari, and M. Forgione, "Learning dynamical systems from quantized observations: A bayesian perspective," *IEEE Transactions on Automatic Control*, vol. 67, no. 10, pp. 5471–5478, 2022.

[4] C. Legaard, T. Schranz, G. Schweiger, J. Drgoňa, B. Falay, C. Gomes, A. Iosifidis, M. Abkar, and P. Larsen, "Constructing neural network based models for simulating dynamical systems," *ACM Comput. Surv.*, vol. 55, no. 11, 2023.

[5] G. Pillonetto, A. Aravkin, D. Gedon, L. Ljung, A. H. Ribeiro, and T. B. Schön, "Deep networks for system identification: A survey," *Automatica*, vol. 171, p. 111 907, 2025.

[6] G. Pillonetto, F. Dinuzzo, T. Chen, G. De Nicolao, and L. Ljung, "Kernel methods in system identification, machine learning and function estimation: A survey," *Automatica*, vol. 50, no. 3, pp. 657–682, 2014.

[7] M. Forgione and D. Piga, "Dynonet: A neural network architecture for learning dynamical systems," *International Journal of Adaptive Control and Signal Processing*, vol. 35, 2021.

[8] M. Forgione and D. Piga, "Continuous-time system identification with neural networks: Model structures and fitting criteria," *European Journal of Control*, vol. 59, pp. 69–81, 2021.

[9] Y. Wang, "A new concept using lstm neural networks for dynamic system identification," in *2017 American Control Conference (ACC)*, 2017, pp. 5324–5329.

[10] M. Kaheni, M. Lippi, A. Gasparri, and M. Franceschelli, "Selective trimmed average: A resilient federated learning algorithm with deterministic guarantees on the optimality approximation," *IEEE Transactions on Cybernetics*, vol. 54, no. 8, pp. 4402–4415, 2024.

[11] N. Bastianello, D. Deplano, M. Franceschelli, and K. H. Johansson, "Robust online learning over networks," *IEEE Transactions on Automatic Control*, vol. 70, no. 2, pp. 933–946, 2025.

[12] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[13] D. Masti and A. Bemporad, "Learning nonlinear state–space models using autoencoders," *Automatica*, vol. 129, p. 109 666, 2021.

[14] G. I. Beintema, M. Schoukens, and R. Tóth, "Deep subspace encoders for nonlinear system identification," *Automatica*, vol. 156, p. 111 210, 2023.

[15] D. Gedon, N. Wahlström, T. B. Schön, and L. Ljung, "Deep state space models for nonlinear system identification," *IFAC-PapersOnLine*, vol. 54, no. 7, pp. 481–486, 2021.

[16] J. L. Paniagua and J. A. López, "Nonlinear system identification using modified variational autoencoders," *Intelligent Systems with Applications*, vol. 22, p. 200 344, 2024.

[17] M. Mejari, M. Forgione, and D. Piga, "Variational autoencoder for the identification of piecewise models," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 4055–4060, 2023.

[18] D. P. Kingma and M. Welling. 2019.

[19] F. Berkenkamp, R. Moriconi, A. P. Schoellig, and A. Krause, "Safe learning of regions of attraction for uncertain, nonlinear systems with gaussian processes," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 4661–4666.

[20] S. M. Richards, F. Berkenkamp, and A. Krause, "The lyapunov neural network: Adaptive stability certification for safe learning of dynamical systems," in *Proceedings of The 2nd Conference on Robot Learning*, A. Billard, A. Dragan, J. Peters, and J. Morimoto, Eds., ser. Proceedings of Machine Learning Research, vol. 87, PMLR, 2018, pp. 466–476.

[21] K. F. Løwenstein, D. Bernardini, L. Fagiano, and A. Bemporad, "Physics-informed online learning of gray-box models by moving horizon estimation," *European Journal of Control*, vol. 74, p. 100 861, 2023.

[22] D. Ushida and E. Konaka, "Model predictive control implementation on neural networks using denoising autoencoder," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2016, pp. 000 149–000 154.

[23] D. Masti, F. Smarra, A. D'Innocenzo, and A. Bemporad, "Learning affine predictors for mpc of nonlinear systems via artificial neural networks," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 5233–5238, 2020.

[24] Y. Cao and R. B. Gopaluni, "Deep neural network approximation of nonlinear model predictive control," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 11 319–11 324, 2020.

[25] A. L. Bishop, J. Z. Zhang, S. Gurumurthy, K. Tracy, and Z. Manchester, "Relu-qp: A gpu-accelerated quadratic programming solver for model-predictive control," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 13 285–13 292.

[26] F. D. Torrisi and A. Bemporad, "Discrete-time hybrid modeling and verification," in *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No. 01CH37228)*, IEEE, vol. 3, 2001, pp. 2899–2904.

[27] J. Katz, I. Pappas, S. Avraamidou, and E. N. Pistikopoulos, "The integration of explicit mpc and relu based neural networks," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 11 350–11 355, 2020.

[28] Y. Liu, R. Tóth, and M. Schoukens, "Physics-guided state-space model augmentation using weighted regularized neural networks," *IFAC-PapersOnLine*, vol. 58, no. 15, pp. 295–300, 2024.

[29] M. Kiss, R. Tóth, and M. Schoukens, "Space-filling input design for nonlinear state-space identification," *IFAC-PapersOnLine*, vol. 58, no. 15, pp. 562–567, 2024.

[30] T. M. Inc., *Machine-learning-based identification of two-tank system*, Natick, Massachusetts, United States, 2024.

[31] Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, 2024. [Online]. Available: `https://www.gurobi.com`.

© 2025 IEEE. Personal use of this material is permitted. DOI: 10.1109/CDC57313.2025.11312089