A Bridge between Dynamical Systems and Machine Learning: Engineered Ordinary Differential Equations as Classification Algorithm (EODECA)

Raffaele Marino[1*], Lorenzo Giambagli[1], Lorenzo Chicchi[1], Lorenzo Buffoni[1], Duccio Fanelli[1]

[1]Dipartimento di Fisica e Astronomia and INFN, Università degli studi di Firenze, Via G. Sansone, Sesto Fiorentino, 50019, FI, Italy.

*Corresponding author(s). E-mail(s): raffaele.marino@unifi.it; Contributing authors: lorenzo.giambagli@unifi.it; lorenzo.chicchi@unifi.it; lorenzo.buffoni@unifi.it; duccio.fanelli@unifi.it;

Abstract

In a world increasingly reliant on machine learning, the interpretability of these models remains a substantial challenge, with many equating their functionality to an enigmatic black box. This study seeks to bridge machine learning and dynamical systems. Recognizing the deep parallels between dense neural networks and dynamical systems, particularly in the light of non-linearities and successive transformations, this manuscript introduces the Engineered Ordinary Differential Equations as Classification Algorithms (EODECAs). Uniquely designed as neural networks underpinned by continuous ordinary differential equations, EODECAs aim to capitalize on the well-established toolkit of dynamical systems. Unlike traditional deep learning models, which often suffer from opacity, EODECAs promise both high classification performance and intrinsic interpretability. They are naturally invertible, granting them an edge in understanding and transparency over their counterparts. By bridging these domains, we hope to usher in a new era of machine learning models where genuine comprehension of data processes complements predictive prowess.

Keywords: Dynamical Systems, Learning methods, Mechanistic Interpretability, Statistical Mechanics

1 Introduction

In an era marked by the exponential growth of data complexity and computational challenges, machine learning (ML) [1, 2] and deep learning (DL) [3, 4] stand as transformative pillars. Their profound impacts are felt across myriad of disciplines, from medicine to finance [5–17]. However, as the sophistication of these tools has increased, so too have the challenges surrounding their interpretability [18]. Deep learning, particularly, operates in ways that are often likened to a *black box*—we can observe its inputs and outputs, but the intricate internal mechanisms that produce these outputs remain elusive. This opacity can undermine trust in ML and DL systems, especially in critical applications where understanding the reasoning behind decisions is imperative.

Interpretability in machine learning denotes our ability to transparently comprehend and trace how algorithms arrive at their conclusions. It serves as a vital check and balance, ensuring models operate logically and fairly, and offering insights into potential flaws or biases. True interpretability goes beyond merely producing reliable results—it seeks to offer clarity on the *why* and *how* behind these outcomes.

In this paper, we embark on an ambitious journey to bridge the realms of machine learning and dynamical systems [19], so to provide novel insight into explicable decision making. While these two domains have seen isolated progress in their respective fields, our work marks a pioneering effort [20, 21], to the best of our knowledge, in truly integrating them. We introduce and construct a unique dynamical system governed by ordinary differential equations (ODEs) that possesses the capability to be trained for classification tasks [22], with a computational complexity of $O(N^2)$, where N is the input size. This is not just an innovation in methodology, it heralds the inception of a new class of algorithms, which we have termed the Engineered Ordinary Differential Equations as Classification Algorithms (EODECAs).

The power of EODECAs lies not just in its ability to classify but in its inherent dynamical system structure. Such a foundation equips us with the unprecedented advantage of tapping into the vast arsenal of tools traditionally reserved for dynamical systems [23]. Consequently, Mechanistic Interpretability [24–26] becomes an intrinsic feature, enabling us not only to bolster the accuracy of our models but also to capitalize on a genuine understanding of the operative processes underpinning the data handling and classification.

At the heart of our methodology lies a unique dynamical system shaped as a neural network. A noticeable feature is that every neuron (or node) within this network operates under the guidance of an ODE that is continuous across both temporal and spatial dimensions. Linear interactions between distinct nodes are in place, as stipulated by an underlying coupling matrix. The system's stationary asymptotic state manifests as a stable attractor of the dynamics, when considering the network as a whole. Different attractors are indeed associated to distinct items to be eventually classified.

Delving deeper, we innovatively introduce the ability to embed, or plant, stable attractors within the dynamical system's phase space. This strategic planting ensures that post reaching its designated attractor in response to an input, the system firmly anchors itself there, sustaining its position indefinitely. Such a characteristic bestows upon the system a layer of trust [27]; the predictability that once a fixed point is attained, deviations are improbable.

But the true prowess of our approach becomes evident when we harness the expansive toolkit of dynamical systems. The ability of the dynamical systems to cope with the assigned classification task leaves a tangible imprint in the extension of the basins of attraction, as sculpted upon training. Further, we gain insight into the concept of invertibility [28], a fundamental byproduct of our design. While traditional deep neural networks scheme can adeptly map inputs to outputs, reversing this process to glean the original input from an output is non-trivial, if not impossible in many instances [29]. This poses challenges, especially when trying to understand the intrinsic behaviors and characteristics of such networks. EODECA, with its design rooted in the principles of dynamical systems, circumvents this limitation, offering not only a robust classification tool but also an invertible architecture that stands out in the realm of machine learning models.

While traditional networks have grappled with opacity, our work with EODECA opens a new chapter in understanding and interpretability in machine learning. With EODECA, we not only address these challenges but also usher in an era where interpretability stands at the forefront of model design.

The paper is structured as follows: in Sec. 2.1, we delve deep into the architecture, principles, and specifics of an EODECA system, by also showing how to embed stable attractors within the dynamical system. Sec. 2.2, sheds light on our experimental setup, offering a comparative analysis between EODECA's performance and traditional models, and discussing detailed findings from our study. More precisely, we present results on a synthetic dataset [30] and then we show the performance of EODECA on two well known benchmark datasets, i.e. MNIST and Fashion MNIST, displaying how we can achieve performance of a multi layer perceptorn (MLP). In Sec. 3 we interpret the results' implications, address any limitations or challenges faced, and propose directions for future research.

2 Results

2.1 Model

We present our learnable autonomous dynamical system on a neural network, composed by N neurons. Overall, the system is described by the following ordinary differential equation:

$$\dot{\vec{x}}(t) = \vec{F}(\vec{x}), \tag{1}$$

where $\vec{x} \in \mathbb{R}^N$, and $\vec{F}(\vec{x}) = -\vec{\nabla} V(\vec{x}(t)) + \beta \mathbf{A} \vec{x}(t)$. We assume $\vec{x}$ having entries of $O(1)$. $\dot{\vec{x}}(t)$ represents the derivative with respect to time of $\vec{x}(t)$. The potential $V(\vec{x}(t)) : \mathbb{R}^N \to \mathbb{R}$ is a scalar field, and $\vec{\nabla}$ denotes the gradient operator. The matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix. It describes the interactions between the nodes of the network. β is a parameter of the system. In the following we will assume a double well potential in the form $V(\vec{x}(t)) = 2\gamma(\vec{x}^2(t) - a^2\vec{\mathbf{1}})^T(\vec{x}^2(t) - a^2\vec{\mathbf{1}})$, where the i-th component of $\vec{x}^2$ is x_i^2, γ and a are two parameters of the dynamical system. The former sets the intensity of the potential, while the latter controls the position of the two symmetric wells, or, stated equivalently, the location of the fixed points of the a-spatial dynamics for the examined system. It should be emphasized that a prototypical double well potential is found to emerge in models relevant to computational neuroscience, as follow the intertwined interaction between distinct families of excitatory and inhibitory neurons [31]. In this respect, the simplified model that we have here formulated unlocks a perspective view full of captivating biomimetic implications.

Notice that the parameter γ is $O(1)$, as the magnitude of the state variables are. It can be indeed treated as a genuinely learnable parameter. At variance, the stabilization of the linear terms requires setting $\beta = \frac{1}{\sqrt{N}}$, to ensure that the effective energy function $\mathcal{E}(\vec{x}(t)) = V(\vec{x}(t)) - \frac{\beta}{2}\vec{x}(t)^T\mathbf{A}\vec{x}(t)$ behaves as a $O(N)$ quantity, when considering the $\mathbf{A}$ entries to be $O(1)$ i.i.d. from a standard Gaussian distribution, as in [32, 33].

The matrix $\mathbf{A}$, as mentioned above, encapsulates all relevant information that pertains to the topology of the underlying network. For reasons that will become transparent in the following we shall assume $\mathbf{A} = \Phi\Lambda\Phi^{-1}$, where $\Phi \in \mathbb{R}^{N\times N}$ and Φ^{-1} is the inverse matrix of Φ. The columns of the matrix Φ are indeed the eigenvectors of matrix $\mathbf{A}$, and the matrix $\Lambda \in \mathbb{R}^{N\times N}$ is diagonal. It contains the eigenvalues of $\mathbf{A}$. For the sake of simplicity we denote with $\vec{\phi}^{(l)}$ the columns of Φ, with $l = 1, .., N$. The choice of dealing with the above decomposition of the coupling matrix echoes the spectral approach to machine learning discussed in [34–37].

The asymptotic attractors, signifying specific configurations that the system can be attracted towards, at large time, are embedded in the matrix Φ, following a procedure that we will hereafter illustrate. Label with $\overset{\rightharpoonup}{\phi}^{(k)}$ the eigenvectors corresponding to our identified stable attractors. Specifically, the $k = 1, \ldots, K$ columns of matrix Φ, where K stands for the total number of classes, as reflecting the intimate complexity of the classification under exam. The entries of the selected vectors $\overset{\rightharpoonup}{\phi}^{(k)}$ can take values $\pm a$ (i.e. the positions of the fixed points of the a-spatial dynamics, $\beta = 0$), namely $\left(\overset{\rightharpoonup}{\phi}^{(k)}\right)_i = \pm a$, $\forall i = 1, ..., N$. By doing so, we liberate a whole reservoir of degrees of freedom to shape a vast gallery of distinct attractors, depending on the specific needs of the problem being inspected. Further, we place the target vectors $\overset{\rightharpoonup}{\phi}^{(k)}$ in the kernel of $\mathbf{A}$. This amounts to say that the corresponding eigenvalues in Λ are identically equal to zero. The above prescriptions implies that $\overset{\rightharpoonup}{\phi}^{(k)}$ with $k = 1, \ldots, K$ are indeed stationary solutions for the spatially coupled dynamics, as it can be proved with a straightforward calculation. As a next step in the story, we need to ensure the stability of the planted attractors. This condition is met by requiring that the eigenvalues of matrix $\mathbf{A}$ lie within the range $(-\infty, \frac{8a^2\gamma}{\beta})$. This conclusion is reached via a standard linear stability analysis of the governing model as described in the Method Section 4.1.

Through careful selection of the residual eigenvalues and the associated eigenvectors' components, we aim at self-consistently shaping the underlying energy landscape, so as to steer the dynamics of the systems towards the deputed attractor. This is the target destination for the item supplied as an input, depending on the class it belongs to. Stated differently, training the network corresponds to shaping the basins of the (asymptotically stable) planted attractors that we have in advance encapsulated in the ruling dynamics.

When it comes to the technical aspects of the training protocol, columns of Φ not directly aligned with the predefined attractors are subjected to a process of stochastic initialization. Similarly, the eigenvalues of matrix $\mathbf{A}$ undergo a randomized initialization, firmly anchored within the boundaries of predefined stability criteria, thereby ensuring the resilient and robust evolution of the system's dynamics. All the columns of Φ not directly involved in the definition of the attractors, as well as the associated eigenvalues, are then treated as learnable parameters during the training process (see Sec. 4.2). A schematic representation of the dynamic system model is displayed in Fig. 1 panel **b**.

2.2 Experimental results

In this section, we focus on presenting the results obtained by applying our model to various benchmarks. The first benchmark examined is a synthetic dataset, meticulously crafted using binarized characters. This dataset, along with its corresponding targets, is illustrated in Figure 1, panel **a**. Utilizing this dataset allows for a clearer and more direct presentation of the obtained results. Subsequently, we will discuss the results achieved on the well-known MNIST [38] and Fashion MNIST [39] datasets, which are established standards for evaluating performance in image classification tasks. Additional details regarding the utilized benchmarks, included the definitions of ϵ_{train} and ϵ_{test}, are available in Sec. 4, specifically in subsections 4.3 and 4.2.

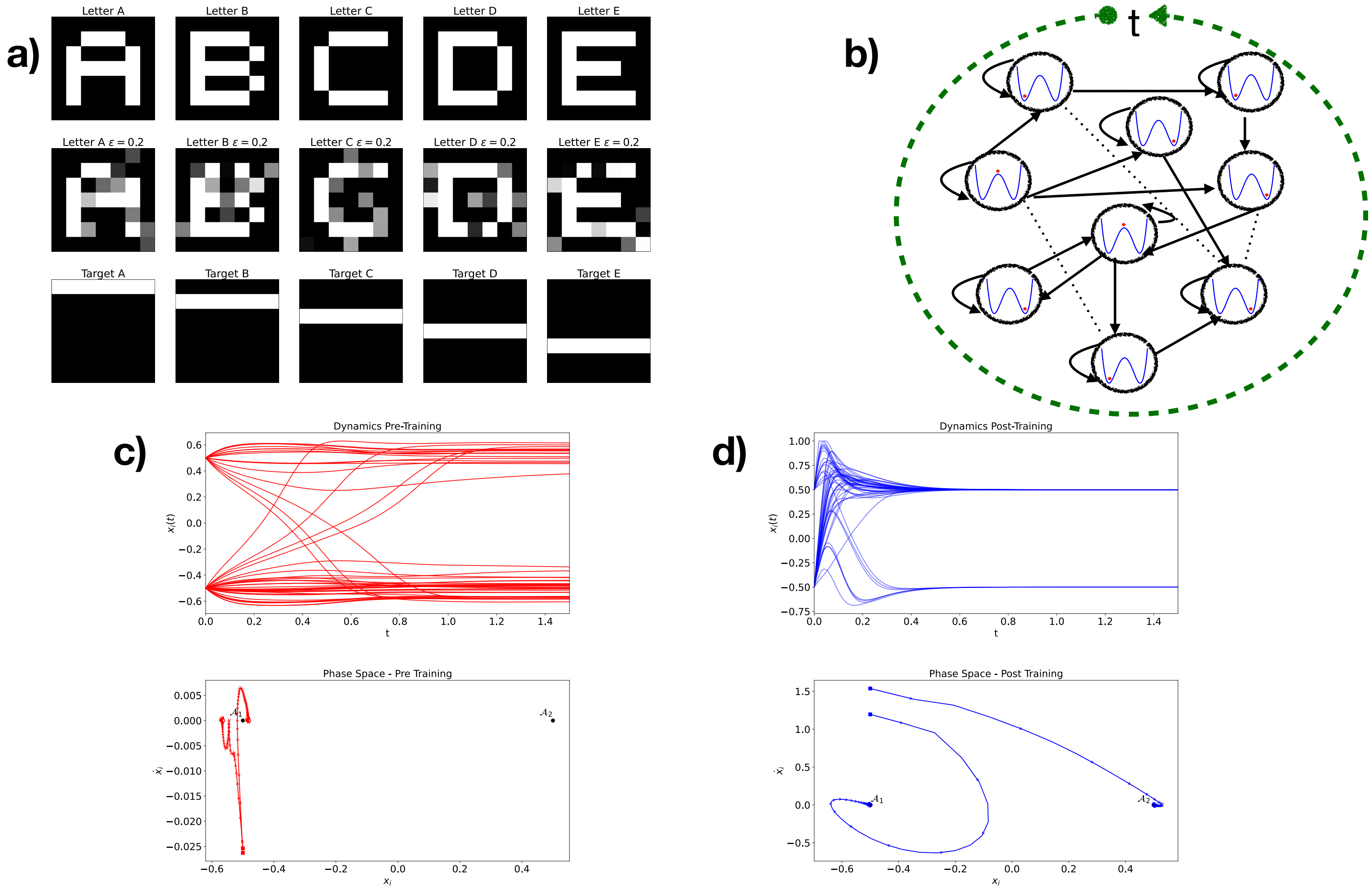

Fig. 1: **a)**:Visual representation of the first five letters of the alphabet $(A - E)$ in a 7×7 grid format. The top row showcases the original, noise-free letters. The middle row displays the same letters but with a noise factor of $\epsilon = 0.2$ applied, introducing slight distortions. The bottom row provides the target representations for each letter, serving as a simplified reference. This depiction allows for a direct comparison between the original, noisy, and target versions of each letter.**b)**: Schematic representation of the dynamic system model employed, illustrating a neural network in which each neuron is uniquely associated with a single pixel and characterized by a double potential well. The red balls indicate the position of pixels in the double well during the dynamics. The adjacency matrix $\mathbf{A}$ highlights the topology of the network, while an outer green circle symbolizes the temporal iterations required to reach the asymptotic steady state.**c)**:*Top panel*-Displays the temporal evolution of all components $x_i(t)$ of the letter E. Here, the components tend to settle on fixed points, however, these do not match the planted ones. *Bottom panel*-Portrays the trajectory in the phase space of two arbitrarily chosen components. The components originate from two distinct starting points, represented by squares, but do not reach the fixed points highlighted by the black dots $\mathcal{A}_1$ and $\mathcal{A}_2$.**d)**:*Top panel*-Reveals the post-training dynamics. Starting from the letter E, the dynamical system readily reaches the designated fixed points. *Bottom panel*-Describes the trajectory of the same components in the phase space after training. In this context, the image pixels are directly transported to their corresponding fixed points. The end points of the trajectories are symbolized by empty circles. Arrows identify the direction of the trajectories. We adopted for the analysis $\mathcal{L} = 2$.

Our analysis, detailed in Fig. 1 panel **c** and **d**, focuses on the system's dynamics evolution before (**c**) and after (**d**) training, using image components such as the letter E. Before training (see upper and lower panels, of **c**), the system's behavior does not accurately reproduce the target of letter E, as components converge to states misaligned with intended fixed points due to influential initialization of matrix $\mathbf{A}$, despite the stability of the letter E's attractor.

Post-training, the optimized matrix $\mathbf{A}$ results in all coordinates consistently aligning with the deputed fixed points (upper panel of **d**). This is further depicted in the phase space in the lower panel of **d**, where post-training trajectories precisely converge to the intended fixed points, marked as $\mathcal{A}_1$ and $\mathcal{A}_2$, contrary to the erratic trajectories observed pre-training panel (lower panel of **c**). The training contributes to shape the basin of the planted attractor in such a way that the supplied item is directed towards the desired destination target.

In our study, a scrupulous analysis was executed, to challenge the effective ability of the scrutinized system to reach the reference asymptotic attractor. As an average measure of the relative distance between the state variable displayed by the system at time t and the final destination target that should be eventually approached, we introduce the Mean Squared Error (MSE) defined as:

$$MSE(t) = \frac{1}{M}\sum_{m=1}^{M}(\vec{x}(t)^{(m)} - \vec{y}^{(m)})^T(\vec{x}(t)^{(m)} - \vec{y}^{(m)}), \tag{2}$$

where M identifies the cardinality of the analyzed data sample, and $\vec{y}$ the m-th target, i.e., the corresponding $\vec{\phi}^{(k)}$. Key variations were unveiled in trained models, attributable to the noise level parameter, ϵ_{train}, manifesting in the MSE behaviors as showcased in Fig. 2 (panels **a** and **b**). Observations reveal that depending on the value of ϵ_{test}, the MSE either approaches zero or stabilizes at non-zero values, underscoring the existence of a dual dynamical phase within the system. A critical insight gleaned from our study is the discernment of trajectories not converging to the attractors. This points to the existence of distinct dynamic phases and attribute significant connotations to the observed transition.

To challenge the actual numericaal convergence, we varied the integration timesteps, Δt, and recovered the results reported in Fig. 2 (panel **c**). We can hence utterly conclude that the reported analysis provides a faithful representation of the underlying continuum dynamical system.

To quantify the system's efficacy to cope with the sought classification task, accuracy was computed as:

$$\overline{\rho}(\vec{x}^*, \vec{y} = \vec{\phi}) = \frac{1}{|\mathcal{D}_{test}|}\sum_{r=1}^{|\mathcal{D}_{test}|}\frac{1}{N}\sum_{i=1}^{N}\delta(x_i^{*(r)}, y_i^{(r)}). \tag{3}$$

Where $\delta(\cdot,\cdot)$ is the Kronecker delta, $\vec{x}^*$ represents the output vector from the model at infinite time ($\pm a$), $\vec{y} = \vec{\phi}$ is the corresponding target to the output vector, and index r identifies the r^{th} instance in the test dataset $\mathcal{D}_{test}$.

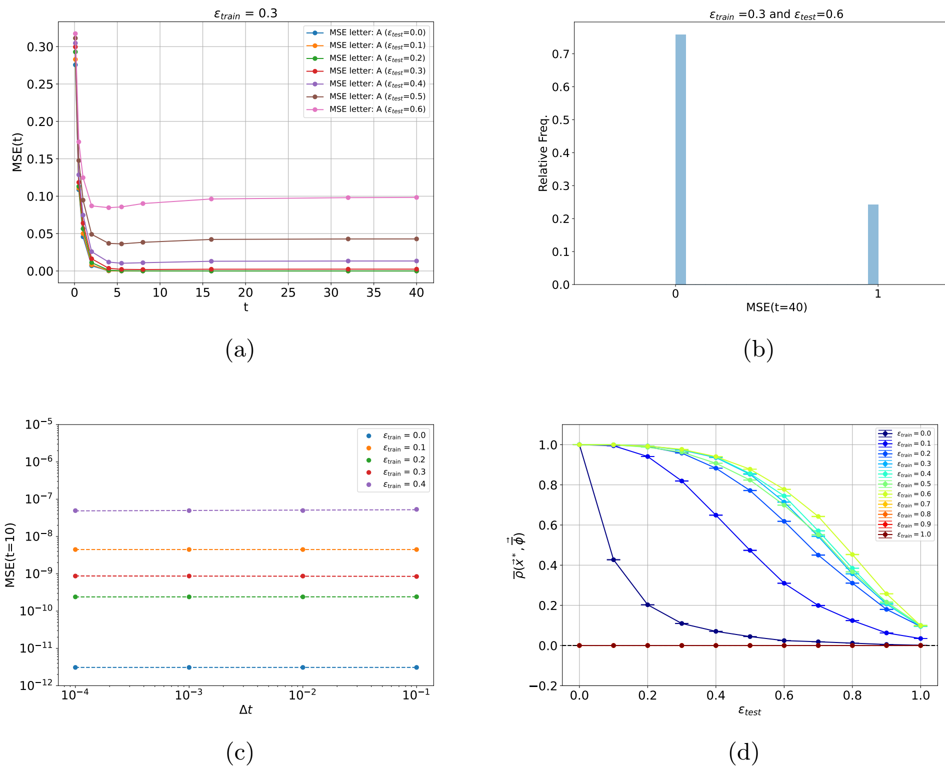

Fig. 2: **(a)** The panel depicts an estimation of the temporal evolution of the Mean Squared Error (MSE) across a sample of $M = 2000$ images A, for various ϵ_{test} values, at a fixed $\epsilon_{train} = 0.3$. For scenarios where $\epsilon_{test} < \epsilon_{train}$, the MSE tends towards zero as time t approaches infinity. Conversely, when $\epsilon_{test} \geq \epsilon_{train}$, the temporal estimate of the MSE increases with added noise. **(b)** A detailed analysis for $t \to \infty$ reveals that the MSE undergoes a first-order transition. In this context, trajectories in the phase space post-training may not converge to the appropriate attractor. This phenomenon is illustrated through the histogram of relative frequencies representing the number of trajectories aligning with the correct attractor. For visual clarity, all non-zero values are designated as 1. **(c)** The plot showcases the MSE at $t = 10$ as a function of the chosen time step Δt for numerical integration, employing the fourth-order Runge-Kutta method. **(d)** The panel represents the accuracy of the model plotted against ϵ_{test} for a range of ϵ_{train} values from 0.0 to 1.0. We adopted for the analysis $\mathcal{L} = 2$.

Analyzed across varied ϵ_{train} and ϵ_{test} values, Fig. 2 (panel **d**) illustrates intriguing performance trends and transitional behaviors, underscoring the nuanced influence of noise levels during training and testing phases.

In essence, our findings illuminate insights into the intricate transition behaviors of the trained dynamical system, while providing, at the same time, an indirect reckoning of the size of the underlying basins of attraction. Taken altogether these latter observations define pivotal avenues for future research exploration.

Upon training, we have obtained a basis of the scanned multi-dimensional space which is tailored to the problem at hand. The basis is formed by the column vectors of matrix Φ. Every supplied item $\vec{\vec{x}}$ can be decomposed by using the aforementioned basis. The set of obtained coefficients, $\vec{\vec{c}} = \Phi^{(-1)}\vec{\vec{x}}$ returns a complete and equivalent representation of the analyzed object $\vec{\vec{x}}$. This framework can be used to probe the model's robustness and vulnerability to external source of disturbance from a different perspective, alternative to perturbing each individual pixel (which amounts to operate with the canonical basis viewpoint). More specifically, we can proceed by perturbing individual eigen-directions - or punctually alter each coefficient $\vec{\vec{c}}$ of the above expansion. In doing so, several pixels get simultaneously modulated by the external noise source, following a pattern of correlated activation that indirectly stems from the accomplished training. In Sec. 4.4 we report the results of this analysis. Remarkably enough only few directions can trigger the system unstable. In fact, we can convincingly show that the vast majority of imposed collective eigen-perturbations do not affect the ability of the system to carry out the assigned classification task. In a world where images are frequently subjected to noise and interference, understanding noise resistance through coefficient perturbation can guide the implementation of resilient designs and/or information compression strategies, ensuring that essential information is preserved despite reductions.

Upon completing the analysis of the synthetic dataset, we can now shift our focus to assess the robustness of EODECA in comparison to the most commonly cited benchmarks in literature: MNIST and Fashion MNIST.

Transitioning from the dataset description to the empirical results on MNIST, we observed promising outcomes with our dynamical approach to classification. Specifically, for the case where $\epsilon_{train} = 0.0$, our system achieved an accuracy of 97.13% over the test set. Considering the inherent simplicity of our method, this performance is quite commendable. For a comparative perspective, it is noteworthy to mention that this accuracy is closely aligned with that of a Multi-Layer Perceptron (MLP) with ReLu activation functions, which boasts an accuracy rate of 98.25%. Such proximate performance metrics underscore the potential and efficacy of our proposed dynamical system, even when juxtaposed against more conventional machine learning architectures. For the Fashion MNIST dataset, our results were illuminating. When set with $\epsilon_{train} = 0.0$, our model, rooted in dynamical systems, garnered an accuracy of 87.01%. Interestingly, this is closely competitive with a conventional Multi-Layer Perceptron (MLP) employing ReLu activation functions, which holds an accuracy of 89.55%. The marginal difference in performance underscores the comparable efficacy of both models, emphasizing that our innovative approach can stand toe-to-toe with established neural network architectures. For contextualizing our findings with the literature, we refer to Sec. 4.5.

Now, as previously shown, we explore the influence of introducing pixel errors into images on the accuracy of classification algorithms. Specifically, our focus lies in investigating the potential expansion of the basin of attraction MNIST and Fashion MNIST. In Table 1, we delineate the behavior of classification accuracy in the MNIST and Fashion MNIST datasets when trained under varying noise conditions — specifically, with $\epsilon_{train} = 0$ (without noise) and $\epsilon_{train} \neq 0$ (with noise). It is palpable from the table that the EODECA model performs consistently better than MLP in the presence of noise across both datasets. For the MNIST dataset, the EODECA's robustness is evidenced by its competitive accuracy relative to the MLP at almost all noise levels.

ϵ_{train}	MNIST		Fashion MNIST	
	EODECA	MLP	EODECA	MLP
0.0	0.9713 ± 0.0017	0.9825 ± 0.0013	0.8701 ± 0.0034	0.8955 ± 0.0030
0.1	0.9780 ± 0.0016	0.9730 ± 0.0016	0.8727 ± 0.0033	0.8825 ± 0.0032
0.2	0.9741 ± 0.0016	0.9641 ± 0.0018	0.8786 ± 0.0033	0.8745 ± 0.0033

Table 1: This table delineates the performance accuracy of two distinct models, EODECA and MLP, when trained under varying levels of noise (ϵ_{train}) an tested in the case $\epsilon_{test} = 0.0$. The accuracy metrics are separately cataloged for two datasets: MNIST and Fashion MNIST. Each cell indicates the model's accuracy, facilitating a comparative assessment of the models' robustness and adaptability in the presence of noise-induced perturbations in the input data during training process.

Further, we delve deeply into exploring the invertibility of our model with respect to the dynamics within EODECA. Invertible machine learning models can be used for data compression, enabling high-fidelity data decompression, critical in fields like medical imaging [40]. They also find applications in generative modeling, where models like normalizing flows leverage invertibility to transform simple distributions into complex ones, useful for synthetic data generation [41]. Another significant application lies in inverse inference, where the model reverses to deduce causes from effects, valuable in regression analysis and fields where causality is essential [42]. Invertibility also aids in model interpretability by maintaining information throughout the network's layers, which can illuminate the model's decision processes [43]. With these applications in mind we set we begin with an image, represented as $\vec{x}(t = 0)$, serving as the initial condition of our dynamics. This image is allowed to evolve in accordance with its inherent dynamical laws. Post-training, it is constructively known that the image concludes its dynamical trajectory at an attractor.

A pivotal question arises at this juncture: Having the output of our classification algorithm $(x(T))$, is it possible to reconstruct the originating image $(x(0))$?

Our findings affirm this possibility. Utilizing a straightforward transformation, $\tau = T - t$ [10, 44], which effectively reverses the dynamics of the system, we can integrate the dynamics backward and recapture the initial image. The evolutionary law to be integrated subsequently becomes $\dot{\vec{x}} = -\vec{F}(\vec{x})$, with the derivative now taken concerning the variable τ.

Figure 3, panel **MNIST**, visually encapsulates this process. Through a demonstrative cartoon, it illustrates the journey from a starting image from the MNIST dataset to the recovered image, obtained via the backward dynamic integration. Figure 3, panel **Fashion MNIST** shows, as an example, the invertibility process for a Fashion MNIST image.

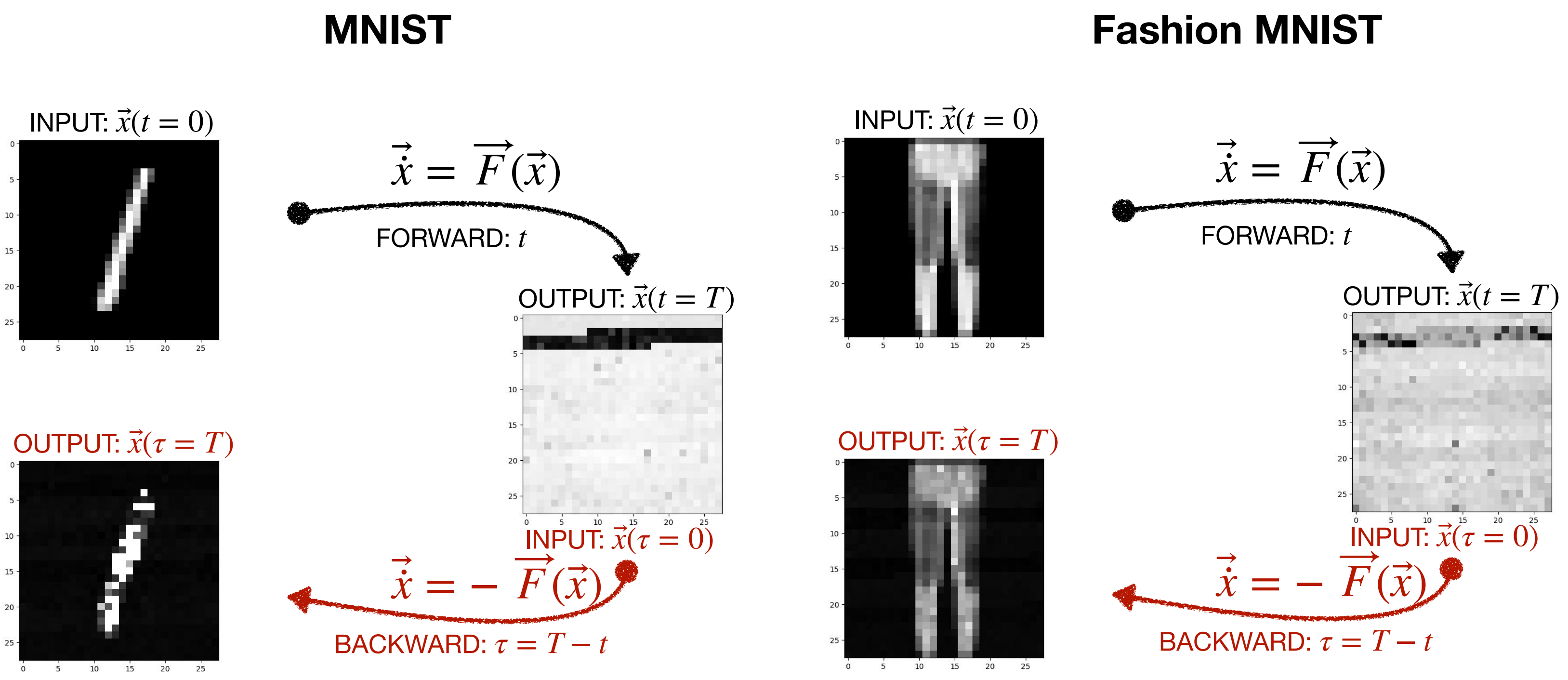

Fig. 3: **MNIST**: The figure delineates the invertibility process deployed within the EODECA model. Given an initial condition $\vec{x}(t=0)$, i.e. an image of the MNIST test set, the dynamical system $\dot{\vec{x}} = \vec{F}(\vec{x})$ evolves to converge onto a designated attractor at a specified time $T = 2.7$. By implementing a temporal inversion in the dynamics, $\dot{\vec{x}} = -\vec{F}(\vec{x})$, and initializing the system with $\vec{x}(\tau=0) = \vec{x}(t=T)$, the system endeavors to reconstruct the inaugural image during the inverse operation. The imperfect reconstruction manifested in the outcome is attributed to numerical errors (loss of information) incurred during integration (forward-backward), utilizing a temporal increment of $\Delta t = 0.01$ in the fourth-order Runge-Kutta numerical integrator. We adopted for the analysis $\mathcal{L} = 10$. **Fashion MNIST**: The figure delineates the invertibility process deployed within the EODECA model. Given an initial condition $\vec{x}(t=0)$, i.e. an image of the FASHION MNIST test set, the dynamical system $\dot{\vec{x}} = \vec{F}(\vec{x})$ evolves to converge onto a designated attractor at a specified time $T = 2.0$. By implementing a temporal inversion in the dynamics, $\dot{\vec{x}} = -\vec{F}(\vec{x})$, and initializing the system with $\vec{x}(\tau=0) = \vec{x}(t=T)$, the system endeavors to reconstruct the inaugural image during the inverse operation. The imperfect reconstruction manifested in the outcome is attributed to numerical errors (loss of information) incurred during integration (forward-backward), utilizing a temporal increment of $\Delta t = 0.0002$ in the fourth-order Runge-Kutta numerical integrator. We adopted for the analysis $\mathcal{L} = 15$.

Invertibility is indeed a distinctive feature of the EODECA dynamics which could be eventually exploited to tackle the above mentioned challenges from a newly crafted perspective.

3 Discussion

EODECA marks a decisive step forward in using dynamical systems theory for robust classification tasks. The foundation of EODECA is the usage of stable attractors within the dynamics, ensuring stability and decisiveness in the classification results, which is particularly resistant to perturbations and noise.

This novel approach integrates the dynamical behavior of the systems into neurons of a network, where each neuron is subjugated by an ordinary differential equation (ODE). EODECA's resemblance to a Recurrent Neural Network (RNN) is profound. It incorporates an architecture where identical neurons are interconnected consistently, akin to layers in a deep network. This structure allows for a seamless flow of information, with each layer's state informed by previous states through consistent weights and connections. By integrating the Euler discretization method, the system described by $\dot{\vec{x}} = \vec{F}(\vec{x})$ can be transformed into $\vec{x}_{n+1} = \vec{x}_n + \Delta t \vec{F}(\vec{x}_n)$. This transformation encapsulates the essence of EODECA as a deep RNN, with depth corresponding to the steps taken in the Euler method for temporal integration.

Within this recurrent framework, the influence of the matrix $\mathbf{A}$ is central, guiding the interaction between neurons from the initial input layer ($n = 0$) across each iterative layer. This structural element underscores EODECA's ability to act recursively, leveraging the Euler integrator to reach the asymptotic state required for classification.

EODECA, in its present configuration, encounters certain boundaries, most apparent when confronting a multitude of classes, where it is imperative to avoid aligning the dimension of the input with the number of classification categories. Overextending in this manner leads to a matrix $\mathbf{A}$ populated with eigenvalues plummeting to zero, relegating it to a null matrix and obfuscating its utility. Additionally, although EODECA aspires to full model interpretability, a significant gap remains in bridging the optimization problems and phase space contraction, crucial for robust interpretability.

Looking ahead, the integration of non-linearities like RELU or sigmoid functions could refine classification performance, warranting further exploration. Moreover, analyzing the bounds of eigenvalues and eigenvectors could enhance training efficiency and algorithmic resilience.

In the realm of adversarial perturbations, which seek to undermine algorithmic integrity, EODECA's adaptability could play an essential role. Its architecture could provide a safeguard against subtle adversarial attacks, critical for applications such as autonomous vehicles where error-proof recognition of road signs is imperative for safety. As suggested by Eshete [27] and Rudin [18], robust classification is essential to counteract such threats.

Future research vistas also unfold in the direction of augmenting EODECA's architecture, possibly integrating elements akin to Convolutional Neural Networks (CNNs). Such thoughtful integrations aim to bolster the performance of EODECA, ensuring an enhancement in its functionality without compromising its intrinsic interpretability as a dynamical system.

In conclusion, EODECA is an embodiment of innovation, promising to push the boundaries of machine learning with its robustness and interpretability as a dynamical system. It stands as a testimonial of the potential convergence of multiple disciplines, poised to forge a new horizon in the field. The continuous exploration into EODECA's applications and capabilities could lead to significant advancements in technology, shaping the future of algorithmic classifications and beyond.

4 Methods

4.1 Linear stability

We start with a general description of our analysis, by recalling the equation (1) for the i-th component of our system:

$$\dot{x}_i(t) = f(x_i(t)) + \beta \sum_{j=1}^{N} A_{ij} x_j(t). \tag{4}$$

In the equation above we have defined $f(x_i(t)) = -4\gamma(x_i^2(t) - a^2)x_i(t)$. Now, let us consider the existence of an eigenvector $\vec{\phi}^{(k)}$, associated with the matrix $\mathbf{A}$ and residing in the kernel of $\mathbf{A}$, namely $\mathbf{A}\vec{\phi}^{(k)} = \vec{0}$. Further, we posit that $\left(\vec{\phi}^{(k)}\right)_i = \pm a$. Hence, $\vec{\phi}^{(k)}$ is a solution for the above system of ODEs.

In order to investigate the linear stability of the above solution, we introduce a perturbation $\delta\vec{x}(t)$ around $\vec{\phi}^{(k)}$. Each perturbed component of $\vec{x}$ is defined as follow: $x_i(t) = \overline{\phi}_i^{(k)} + \delta x_i(t)$, with $\delta x_i(t)$ representing a small deviation from the state $\overline{\phi}_i^{(k)}$.

The model, represented by equation (4), is then linearized around $\vec{\phi}^{(k)}$, leading to the following expression for each component:

$$\begin{aligned} \delta\dot{x}_i(t) = f(\overline{\phi}_i^{(k)}) + f'(\overline{\phi}_i^{(k)})\delta x_i(t) + \\ \beta \sum_{j=1}^{N} A_{ij}\overline{\phi}_j^{(k)} + \beta \sum_{j=1}^{N} A_{ij}\delta x_j(t) + O(\delta x_i^2(t)), \end{aligned} \tag{5}$$

where $'$ identifies the derivative respect to the variable $x_i(t)$. Higher-order terms in $\delta x_i(t)$ are neglected in this treatment. By utilizing the initial assumptions, i.e., $f(\vec{\phi}^{(k)}) = \vec{0}$ and $\mathbf{A}\vec{\phi}^{(k)} = \vec{0}$, equation (5) can be further simplified to:

$$\delta\dot{x}_i(t) = f'(\overline{\phi}_i^{(k)})\delta x_i(t) + \beta \sum_{j=1}^{N} A_{ij}\delta x_j(t) \tag{6}$$

To proceed with the linear stability analysis, we aim to express $\delta x_i(t)$ in the basis of eigenvectors of $\mathbf{A}$. In this context, $\delta x_i(t) = \sum_\alpha c_\alpha(t)\phi_i^{(\alpha)}$.

By substituting $\delta x_i(t)$ with the expression above, we obtain an equation for the evolution of coefficients $c_\alpha(t)$. This equation is given by:

$$\sum_\alpha \phi_i^{(\alpha)} \left\{ \dot{c}_\alpha(t) - \left(f'(\pm a) + \beta\lambda^{(\alpha)} \right) c_\alpha(t) \right\} = 0, \; \forall\alpha. \tag{7}$$

The condition for a stable solution of the differential equation $\dot{c}_\alpha = (f'(\pm a) + \beta\lambda^{(\alpha)})c_\alpha$ is thus provided by $\lambda_i^{(\alpha)} < \frac{8a^2\gamma}{\beta}$, $\forall\alpha$. From the solution of the differential equation, we can immediately observe that if $\lambda^{(\alpha)} < \frac{8a^2\gamma}{\beta}$ $\forall\alpha$ then $\vec{\phi}^{(k)}$ is a stable attractor of the dynamics, meaning that perturbations decay exponentially with time. Conversely, if there exist at least one $\lambda^{(\alpha)} > \frac{8a^2\gamma}{\beta}$ then $\vec{\phi}_i^{(k)}$ the fixed point becomes unstable, as well as the whole attractor, as a small perturbation grows exponentially with time. This allows us to constrain the eigenvalues of the matrix $\mathbf{A}$ to only be in the range $(-\infty, \frac{8a^2\gamma}{\beta})$.

In computational realms, we often encounter functions characterized by steep slopes, indicating rapid changes over minuscule intervals. One prominent exemplar is our $V(\vec{x}(t))$. As time progresses, the evolution of our system can cause $\vec{x}(t)$ to undergo swift and, at times, extreme fluctuations. Such abrupt variations might thrust $\vec{x}(t)$ into zones where $V(\vec{x}(t))$ loses numerical stability. To counteract this, we employ a safeguard: each component, $x_i(t)$, of $\vec{x}(t)$ is meticulously *clipped* within the bounds of $[-\mathcal{L}a, \mathcal{L}a]$, where $\mathcal{L}$ can be chosen by the costumer. This measure ensures that $\vec{x}(t)$ consistently operates within numerically stable territories for $\nabla V(\vec{x}(t))$.

In conclusion, we have examined the linear stability of our model, grounded in assumptions regarding the eigenvectors of the matrix $\mathbf{A}$ and the properties of perturbations around stable equilibrium points. The final stability condition yields an upper limit for the eigenvalues λ_i, ensuring the system maintains a stable solution over time.

4.2 Training

Under the above conditions on the stability, we can construct the matrix $\mathbf{A}$ by setting the eigenvectors associated with the attractors in the columns of the matrix Φ and the corresponding eigenvalues in Λ to zero. All columns of Φ that do not correspond to planted attractors are initialized randomly. Similarly, the eigenvalues of $\mathbf{A}$ are also initialized randomly, but they must satisfy the constraints described above. It is essential to highlight that, since the planted eigenvectors are by definition in the kernel of $\mathbf{A}$, they can also be constructed as linear combinations of orthonormal vectors of the kernel of $\mathbf{A}$.

Without loss of generality, we fix the parameters a at 0.5 and let γ be tunable. Given the inherent challenges in manual selection of the matrix Φ and the Λ weights not associated to planted attractors, achieving accurate classification becomes a daunting task. Therefore, to empower our network with the ability to classify effectively, we resort to a learning methodology.

Moving onto the model training, the learnable parameters encompass the unplanted eigenvectors of Φ. The eigenvalues of Λ corresponding to these columns are also included, while those corresponding to the dynamics' stable points are set to zero by construction and they are fixed during the training process. The parameter space for optimization resides in $\mathbb{R}^{((N-K)\times(N-K))+N-K+1}$, where K denotes the number of classes in our classification problem, and the singular dimension relates to the parameter γ.

The objective is to minimize the loss function $\mathcal{L} = \frac{1}{B}\sum_{j=1}^{B}(\vec{x}^{*(j)} - \vec{y}^{(j)})^T(\vec{x}^{*(j)} - \vec{y}^{(j)})$, with B as the size of the sample. $\vec{x}^*$ is the value of $\vec{x}^* = \vec{x}(T)$, when T is large enough, i.e. the dynamical system in (1) reaches its stationary state. In practice, we take a dataset $\mathcal{D} = (\vec{x}, \vec{y})^{(j)\in[1,\dots,|\mathcal{D}|]}$ of size $|\mathcal{D}|$, where $\vec{x}^{(j)}$ is an input datum, and $\vec{y}^{(j)}$ represents the target, mapped into one of our attractor $\vec{\phi}^{(k)}$ of the dynamics (with $k = 1, \dots, K$). These are sampled i.i.d. from their joint distribution $\mathbf{P}(\vec{x}, \vec{y})$. This dataset is then split into training and test sets. In other words, we give as initial condition to our dynamical system the input datum, i.e., $\vec{x}(0) = \vec{x}^{(j)}$, and we let evolve the system for large enough a time T. Reached the time T, the value of $\vec{x}^{*(j)} = \vec{x}^{(j)}(T)$ is used for optimizing the loss function. The optimization process can be facilitated using a simple algorithm, for which we opt for Stochastic Gradient Descent (SGD), i.e. Adam [45]. Once optimal weight configuration is attained via SGD, we can proceed to evaluate the dynamical system's performance for classifications on test set elements. Importantly, these test set elements are a subset of the dataset $\mathcal{D}$ that the system has never encountered during the training process. All numerical integrations for our dynamical process were carried out using the fourth-order Runge-Kutta method [46]. This method is employed for its precision, having a truncation error of $O(\Delta t^4)$, where Δt is the time step of the integration. However, Euler discretization [46], can also be used during training, achieving comparable performance for all trained models. We recall that the Euler truncation error is $O(\Delta t)$. Unless explicitly stated otherwise, the fourth-order Runge-Kutta method is implied throughout the manuscript. The time step, Δt, utilized for the integration was set at 0.1, while the time of integration T was set to 5.5.

In the methodology of our study, we incorporated an element of noise during the training phase to rigorously assess the robustness of our model. As detailed in Sec. 4.3, our training dataset was intentionally corrupted with uniform noise, a strategy meticulously crafted to simulate a range of potential disturbances and uncertainties, thereby enhancing the model's resilience and adaptive capacities. This deliberate induction of noise is parameterized by ϵ_{train}, a variable that quantifies the level of corruption infused into the model during the training process.

For our analyses, presented in the Sec. 2, we utilized test sets subject to variable levels of corruption to scrutinize the model's performance and reliability under diverse conditions. The magnitude of the noise introduced in this phase is denoted by ϵ_{test}, serving as a metric to gauge the extent of deviation introduced into the test set during the evaluative analysis. This systematic approach, underscored by clearly defined parameters, facilitates a comprehensive and nuanced understanding of the model's robustness and adaptability in navigating and mitigating the impact of noise during its operational functionality.

4.3 Letter dataset

In this section, we describe the synthetic dataset, designed using binarized characters, with values $\{0, 255\}$ on each pixel, to match the specificities of our experimental needs. To the best of our knowledge, such a dataset bridges the dichotomy of simplicity and richness. It not only enables low computational overhead but also morphs into a complex repository when noise elements are introduced. Each character in this curated collection is encapsulated within a 7×7 pixel grid depicted in grayscale. This framework ensures the dual advantage of safeguarding essential geometric intricacies and facilitating computational efficiency.

Appreciating the seldom pristine nature of data in real-world settings, we have integrated uniform noise into arbitrary pixels, thereby mirroring practical scenarios and endowing the dataset with layers of realistic complexity. By adjusting the noise parameter $\epsilon \in [0, 1]$, we wield control over its intensity, setting the stage for systematic inquiries into the model's resilience against noise. Quantitatively, the number of random corrupted pixels, represented as ζ, is deduced as $\zeta = \epsilon \times N$, where N denotes the totality of pixel constituents in an image. The noise on each chosen pixel is uniformly sampled from $[0, 255]$. This methodology yields an intensive parameter, steadfast even as N approaches infinity, underpinning the scalability of our findings. An illustrative snapshot of this model is presented in Fig. 1, panel **a**. This visual representation unfurls the initial five letters of the alphabet $A - E$ structured within a 7×7 grid layout. The topmost row delineates the pristine, noise-void version of the letters. In contrast, the intermediate row paints these letters, albeit with an overlay of noise, marked by a factor $\epsilon = 0.2$, inducing discernible perturbations. The concluding row represent the target visualizations for every individual letter. Such a portrayal promotes a side-by-side juxtaposition between the archetypal, noise-tainted, and target avatars of each letter, offering readers a comprehensive insight into the transformation spectrum, of our $K = 5$ classification problem.

4.4 Perturbations of coefficients in an image

An image, or vector $\vec{x}$, can be expressed as the product of the matrix Φ and the coefficient vector $\vec{c}$, as outlined in Sec. 2.1. A central question is whether certain coefficients withstand perturbations and the extent of perturbation they can tolerate before the trained model can no longer correctly classify such images. Essentially, we aim to understand a model's robustness in light of variations in the $\vec{c}$ coefficients of the provided input image. Moreover, we sense the size of the basins of attraction, post training, by using a correlated version of noise that is self-consistently shaped by the learning protocol. To delve into this, following common approaches across various disciplines, we perturb one coefficient at a time. This allows us to identify coefficients that may be inherently more robust than others. Fig. 4 displays this analysis in 3D: it examines the robustness of each image coefficient against a Gaussian perturbation $\mathcal{N}(0, \sigma)$ applied individually. The MSE, calculated in a steady state, serves as a benchmark metric. We observe that, for a model trained with $\epsilon_{train} = 0.3$, many coefficients are resilient to individual perturbations. This hints at the potential to explore multiple and cumulative perturbations, as depicted in Fig.5. The provided figure highlights in

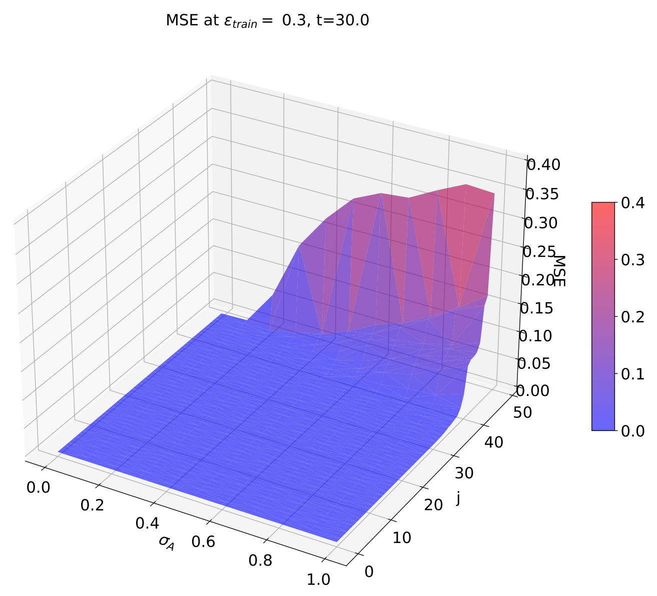

Fig. 4: In the three-dimensional figure presented, the x-axis delineates the variance, σ_A, of the Gaussian perturbation applied to a specific component of the coefficient vector $\vec{c}^{(A)}$ corresponding to the A letter, the y-axis displays the index j of the perturbed component from the vector $\vec{c}^{(A)}$, and the z-axis represents the sample Mean Squared Error (MSE) value. Notice that the original image can be written as $\vec{x}^{(A)} = \Phi\vec{c}^{(A)}$, where Φ signifies the transformation matrix. Consequently, by leveraging the inverse of Φ, we can derive the coefficient vector $\vec{c}^{(A)}$. In this representation, we introduce a perturbation exclusively to one component of the vector $\vec{c}^{(A)}$ at a given instance and compute the corresponding MSE. The indices on the y-axis are systematically arranged based on the ascending magnitude of the MSE, thereby illustrating the differential impact of perturbations across various components of $\vec{c}^{(A)}$ on the overall error in the reconstructed image. We adopted for the analysis $\mathcal{L} = 2$.

fact the robustness of different dynamical models, each trained with a specific ϵ_{train}, against accumulated perturbations on a fraction of coefficients. The main objective of this representation is to determine the threshold beyond which the cumulative effect of these perturbations impairs the accurate classification of a reconstructed image, particularly referring to the amplitude of the perturbation, explicitly its Gaussian variance, for MSE values below 5%. From an initial analysis, it is clear that in the absence of perturbations, all coefficients naturally appear robust. However, introducing noise

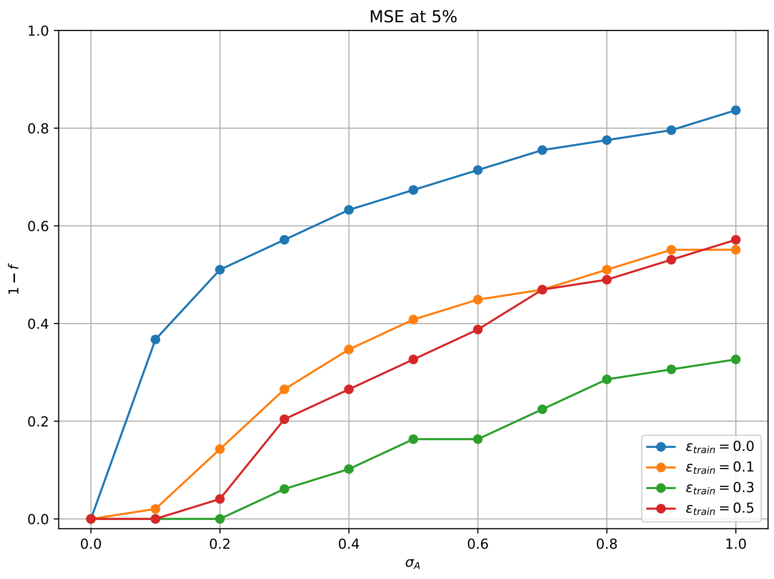

Fig. 5: The figure illustrates the fraction of components that cannot be cumulatively added due to perturbations on each individual component, in relation to the amplitude of the perturbation, specifically the variance of the Gaussian, for MSE values less than 5%, for the letter A. The primary objective is to ascertain the threshold beyond which the cumulative effect of these perturbations impedes the accurate classification of a reconstructed image. This behavior is depicted across various ϵ_{train} values, representing models trained with different noise levels. A discernible observation from the figure suggests the existence of an optimal noise level up to which the perturbations remain inconsequential for the classification task. In this specific instance, the optimal value is identified as $\epsilon_{train} = 0.3$. We adopted for the analysis $\mathcal{L} = 2$.

significantly alters this stability perception. For instance, with $\epsilon_{train} = 0.0$, a moderate perturbation (equivalent to $\sigma_A = 0.1$) affecting 63% of the coefficients results in a significant MSE increase, pushing it beyond the 5% threshold. In contrast, this behavior is attenuated or, in some cases, nearly absent in models trained with $\epsilon_{train} > 0.0$. A crucial observation clearly emerges from the figure: there exists an optimal noise level up to which perturbations remain essentially inconsequential for the classification task. In this particular instance, the optimal value is pinpointed as $\epsilon_{train} = 0.3$. This suggests that training with images subjected to a certain noise level might indeed bolster the resistance of coefficients against various perturbations.

4.5 Comparison with literature MNIST and Fashion MNIST

Moving forward with our comparative analysis, the EODECA method, intrinsically interpretable as a dynamical system, holds a distinctive edge over other interpretable methodologies currently documented in literature. As an illustrative benchmark, the

MNIST dataset has been subjected to various analytical techniques, including the Hopfield model, both with and without the *dreaming* approach [47, 48]. However, these methods have, at best, reached performance metrics below the 70% threshold, and with a supervising approach, where it is turned Hebb's into a genuine learning rule, can reach a 94% of accuracy. Furthermore, when MNIST was probed using a restricted Boltzmann machine, even with the incorporation of dreaming strategies, the performance consistently lingered below 75% [49]. Such comparisons underline a salient observation: our approach, despite its elemental nature relative to other proposals in the literature, emerges not only as the most parsimonious but also as the most efficacious. This duality of simplicity and performance underscores the promising potential of our proposed dynamical system in the realm of interpretable machine learning.

For the Fashion MNIST dataset, when contextualizing our findings with the literature, the comparative strengths of our model become apparent. The Hopfield Model, even when enhanced with strategies like dreaming, achieves a performance just shy of 63%, with a supervising approach, where it is turned Hebb's into a genuine learning rule, it can reach the 84% [49]. This is noteworthy when juxtaposed with our model's 87.01% accuracy. Our approach not only outperforms the Hopfield Model but does so with a more streamlined and simpler structure. This underscores the efficiency and potential of our system in relation to established methods.

Supplementary information. No Supplementary information.

Acknowledgments. This work is supported by #NEXTGENERATIONEU (NGEU) and funded by the Ministry of University and Research (MUR), National Recovery and Resilience Plan (NRRP), project MNESYS (PE0000006) ”A Multiscale integrated approach to the study of the nervous system in health and disease” (DN. 1553 11.10.2022).

Declarations

The numerical codes used in this study and the data that support the findings are available from the corresponding author upon request. The authors declare no competing financial interests.

References

[1] Bishop, C.M.: Pattern Recognition and Machine Learning, 5th Edition. Information science and statistics, (Springer, 2007)

[2] Shalev-Shwartz, S., Ben-David, S.: Understanding Machine Learning: From Theory to Algorithms, (Cambridge university press, 2014)

[3] LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. nature **521**(7553), 436–444 (2015)

[4] Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning, (MIT press, 2016)

[5] Esteva, A., Robicquet, A., Ramsundar, B., Kuleshov, V., DePristo, M., Chou, K., Cui, C., Corrado, G., Thrun, S., Dean, J.: A guide to deep learning in healthcare. Nature medicine **25**(1), 24–29 (2019)

[6] Ching, T., Himmelstein, D.S., Beaulieu-Jones, B.K., Kalinin, A.A., Do, B.T., Way, G.P., Ferrero, E., Agapow, P.-M., Zietz, M., Hoffman, M.M., *et al.*: Opportunities and obstacles for deep learning in biology and medicine. Journal of The Royal Society Interface **15**(141), 20170387 (2018)

[7] Razzak, M.I., Naz, S., Zaib, A.: Deep learning for medical image processing: Overview, challenges and the future. Classification in BioApps: Automation of Decision Making, 323–350 (2018)

[8] Heaton, J.B., Polson, N.G., Witte, J.H.: Deep learning for finance: deep portfolios. Applied Stochastic Models in Business and Industry **33**(1), 3–12 (2017)

[9] Sezer, O.B., Gudelek, M.U., Ozbayoglu, A.M.: Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. Applied soft computing **90**, 106181 (2020)

[10] Marino, R., Macris, N.: Solving non-linear kolmogorov equations in large dimensions by using deep learning: a numerical comparison of discretization schemes. Journal of Scientific Computing **94**(1), 8 (2023)

[11] Marino, R.: Learning from survey propagation: a neural network for max-e-3-sat. Machine Learning: Science and Technology **2**(3), 035032 (2021)

[12] Giambagli, L., Buffoni, L., Carletti, T., Nocentini, W., Fanelli, D.: Machine learning in spectral domain. Nature communications **12**(1), 1330 (2021)

[13] Buffoni, L., Civitelli, E., Giambagli, L., Chicchi, L., Fanelli, D.: Spectral pruning of fully connected layers. Scientific Reports **12**(1), 11201 (2022)

[14] Chicchi, L., Giambagli, L., Buffoni, L., Carletti, T., Ciavarella, M., Fanelli, D.: Training of sparse and dense deep neural networks: Fewer parameters, same performance. Physical Review E **104**(5), 054312 (2021)

[15] Chicchi, L., Fanelli, D., Giambagli, L., Buffoni, L., Carletti, T.: Recurrent spectral network (rsn): Shaping a discrete map to reach automated classification. Chaos, Solitons & Fractals **168**, 113128 (2023)

[16] Baldassi, C., Lauditi, C., Malatesta, E.M., Perugini, G., Zecchina, R.: Unveiling the structure of wide flat minima in neural networks. Physical Review Letters **127**(27), 278301 (2021)

[17] Lucibello, C., Pittorino, F., Perugini, G., Zecchina, R.: Deep learning via message passing algorithms based on belief propagation. Machine Learning: Science and

Technology **3**(3), 035005 (2022)

[18] Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature machine intelligence **1**(5), 206–215 (2019)

[19] Ott, E.: Chaos in Dynamical Systems, (Cambridge University Press, 2002)

[20] Prince, S.J.: Understanding Deep Learning, (MIT press, 2023)

[21] Strogatz, S.H.: Nonlinear Dynamics and Chaos with Student Solutions Manual: With Applications to Physics, Biology, Chemistry, and Engineering, (CRC press, 2018)

[22] Kotsiantis, S.B., Zaharakis, I., Pintelas, P., *et al.*: Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer engineering **160**(1), 3–24 (2007)

[23] Vulpiani, A., Cecconi, F., Cencini, M.: Chaos: from Simple Models to Complex Systems vol. 17, (World Scientific, 2009)

[24] Liu, Z., Michaud, E.J., Tegmark, M.: Omnigrok: Grokking beyond algorithmic data. arXiv preprint arXiv:2210.01117 (2022)

[25] Liu, Z., Kitouni, O., Nolte, N.S., Michaud, E., Tegmark, M., Williams, M.: Towards understanding grokking: An effective theory of representation learning. Advances in Neural Information Processing Systems **35**, 34651–34663 (2022)

[26] Conmy, A., Mavor-Parker, A.N., Lynch, A., Heimersheim, S., Garriga-Alonso, A.: Towards automated circuit discovery for mechanistic interpretability. arXiv preprint arXiv:2304.14997 (2023)

[27] Eshete, B.: Making machine learning trustworthy. Science **373**(6556), 743–744 (2021)

[28] Behrmann, J., Grathwohl, W., Chen, R.T., Duvenaud, D., Jacobsen, J.-H.: Invertible residual networks. In: International Conference on Machine Learning, pp. 573–582 (2019). PMLR

[29] Song, Y., Meng, C., Ermon, S.: Mintnet: Building invertible neural networks with masked convolutions. Advances in Neural Information Processing Systems **32** (2019)

[30] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)

[31] Zankoc, C., Biancalani, T., Fanelli, D., Livi, R.: Diffusion approximation of the stochastic wilson-cowan model. Chaos, Solitons & Fractals **103**, 504–512 (2017)

[32] Sherrington, D., Kirkpatrick, S.: Solvable model of a spin-glass. Physical review letters **35**(26), 1792 (1975)

[33] Panchenko, D.: The Sherrington-kirkpatrick Model, (Springer Science & Business Media, 2013)

[34] Giambagli, L., Buffoni, L., Carletti, T., Nocentini, W., Fanelli, D.: Machine learning in spectral domain. Nature Communications **12**(1) (2021)

[35] Buffoni, L., Civitelli, E., Giambagli, L., Chicchi, L., Fanelli, D.: Spectral pruning of fully connected layers. Scientific Reports **12**(1) (2022)

[36] Chicchi, L., Giambagli, L., Buffoni, L., Carletti, T., Ciavarella, M., Fanelli, D.: Training of sparse and dense deep neural networks: Fewer parameters, same performance. Phys. Rev. E **104**, 054312 (2021)

[37] Chicchi, L., Fanelli, D., Giambagli, L., Buffoni, L., Carletti, T.: Recurrent spectral network (rsn): Shaping a discrete map to reach automated classification. Chaos, Solitons & Fractals **168**, 113128–113128 (2023)

[38] Deng, L.: The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine **29**(6), 141–142 (2012)

[39] Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747 (2017)

[40] Kingma, D.P., Dhariwal, P.: Glow: Generative flow with invertible 1x1 convolutions. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 31 (2018)

[41] Rezende, D., Mohamed, S.: Variational inference with normalizing flows. In: Bach, F., Blei, D. (eds.) Proceedings of the 32nd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 37, pp. 1530–1538. PMLR, Lille, France (2015)

[42] Ardizzone, L., Kruse, J., Rother, C., Köthe, U.: Analyzing inverse problems with invertible neural networks. In: International Conference on Learning Representations (2019)

[43] Jacobsen, J.-H., Smeulders, A.W.M., Oyallon, E.: i-revnet: Deep invertible networks. In: International Conference on Learning Representations (2018)

[44] Han, J., Jentzen, A., E, W.: Solving high-dimensional partial differential equations using deep learning. Proceedings of the National Academy of Sciences **115**(34), 8505–8510 (2018)

[45] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, arXiv:1412.6980 (2015)

[46] Butcher, J.C.: Numerical Methods for Ordinary Differential Equations, (John Wiley & Sons, 2016)

[47] Belyaev, M.A., Velichko, A.A.: Classification of handwritten digits using the hopfield network. IOP Conference Series: Materials Science and Engineering **862**(5), 052048 (2020)

[48] Leonelli, F.E., Agliari, E., Albanese, L., Barra, A.: On the effective initialisation for restricted boltzmann machines via duality with hopfield model. Neural Networks **143**, 314–326 (2021)

[49] Alemanno, F., Aquaro, M., Kanter, I., Barra, A., Agliari, E.: Supervised hebbian learning. Europhysics Letters **141**(1), 11001 (2023)