A Machine Learning Model for Humidity Estimation Based on Physics-Informed Dimensionality Reduction

Nonlinear Dyn
https://doi.org/10.1007/s11071-025-11771-3

RESEARCH

A Machine Learning Model for Humidity Estimation Based on Physics-Informed Dimensionality Reduction

Alessandro Licciardi · Sara Bernardi · Marco Pizzi · Paolo Begnamino · Lamberto Rondoni

Received: 30 May 2025 / Revised: 28 August 2025 / Accepted: 31 August 2025

Abstract Accurate and efficient humidity estimation is critical for various applications, particularly with the rise of IoT devices and smart sensors where computational resources are limited. A common issue with many conventional humidity sensors is their slow response dynamics, which restricts their performance in applications requiring rapid, real-time data. This work introduces a novel Machine Learning algorithm for fast humidity estimation, based on analyzing the voltage discharge dynamics across microelectrodes, computationally frugal yet accurate estimation method suitable for resource-constrained environments. We propose a Physics-Informed Dimensionality Reduction (PIDR) methodology that leverages an underlying physical model – specifically, anomalous diffusion governing the electric discharge between microelectrodes – to extract low-dimensional, physically meaningful features from high-dimensional sensor time series data. Neural Parameter Estimation (NPE), trained effectively on synthetically augmented data guided by limited experimental observations, maps the voltage discharge curves to the anomalous diffusion parameters of the physical model. These parameters, representing a low dimensional physical space, are then fed alongside temperature readings into a compact Artificial Neural Network (ANN) for final humidity prediction. This two-stage, physics-aware architecture significantly reduces model complexity. Experimental results demonstrate the effectiveness of the PIDR approach, achieving high prediction accuracy while demanding significantly less computational effort and training data than traditional parameter estimation techniques or purely data-driven deep learning models applied to raw data. Our study highlights the successful integration of physical principles with machine learning for developing efficient, interpretable, and robust AI solutions tailored for smart sensors and sustainable IoT applications.

Keywords Machine Learning · Humidity Estimation · Physics-Informed Dimensionality Reduction

A. Licciardi · S. Bernardi · L. Rondoni
Department of Mathematical Sciences, Politecnico di Torino, Cso. Duca degli Abruzzi 24, 10129 Torino, Italy
e-mail: sara.bernardi@cnr.it

L. Rondoni
e-mail: lamberto.rondoni@polito.it

A. Licciardi (✉) · L. Rondoni
INFN, Sezione di Torino, Torino 10125, Italy
e-mail: alessandro.licciardi@polito.it

M. Pizzi · P. Begnamino
Research Department, Eltek S.p.A., Strada Valenza 7, 15033 Casale Monferrato, Italy
e-mail: m.pizzi@eltekgroup.it

P. Begnamino
e-mail: p.begnamino@eltekgroup.it

S. Bernardi
Institute of Atmospheric Sciences and Climate, National Research Council of Italy, Corso Fiume 4, 10133 Torino, Italy

Published online: 11 September 2025

1 Introduction

The precise and fast estimation of air humidity is essential across a multitude of applications, ranging from environmental monitoring and agricultural management to sophisticated industrial process control [1–3]. While the proliferation of low-cost humidity sensors offers widespread accessibility, these devices often struggle with inherent limitations such as nonlinear responses, calibration drift, and susceptibility to environmental factors, necessitating advanced signal processing techniques [4,5]. Capacitive sensors, favored for their stability and cost-effectiveness [6], measure humidity-induced changes in dielectric properties, often monitored via electrical characteristics like capacitor discharge times [7]. A significant drawback of most humidity sensing technologies is their slow response – often dictated by the diffusion-limited interaction of moisture with the sensing element – which restricts their utility in dynamic scenarios. Machine learning has emerged as a powerful tool to enhance the accuracy of such sensors, compensating for errors and linearizing outputs [5,6]. However, deploying sophisticated ML models directly on resource-constrained sensor hardware presents challenges regarding computational efficiency and the need for large training datasets.

This paper investigates the feasibility of a novel humidity sensing approach based on analyzing the voltage discharge characteristics of a capacitor connected across a microelectrode gap immersed in the air sample. The core principle relies on the fact that the discharge dynamics are modulated by the conductivity of the air, which is strongly influenced by humidity. Our primary objective is to develop a predictive model that estimates humidity H from the observed time-dependent voltage discharge curve $\tilde{u}$, potentially incorporating ambient temperature T as conditioning information. Crucially, we aim to design a solution adhering to strict constraints relevant for smart sensor applications: the model must perform reliably even when trained on limited experimental data and must be computationally frugal, suitable for deployment on low-power embedded hardware.

To address these challenges, we propose a physics-informed machine learning framework centered around the concept of **Physics-Informed Dimensionality Reduction**. Instead of directly mapping the high-dimensional voltage time series to humidity using a potentially complex model, our approach first projects the observational data onto a low-dimensional latent space defined by the parameters of an underlying physical model. Specifically, we model the voltage decay using principles of anomalous diffusion within the microgap. Indeed, the experimentally-informed mathematical modeling approach proposed in [8] has proven effective in characterizing the transport properties of electrical discharge, enabling the classification of fluids with varying conductive and insulating properties, including mixtures containing water. We then employ Neural Parameter Estimation (NPE), a probabilistic machine learning technique for parameter estimation [9,10], to efficiently infer the key physical parameters governing the observed discharge curve from the input time series $\tilde{u}$. This NPE step, enhanced by training on synthetic data generated via a Gaussian Mixture Model (GMM) prior, effectively reduces the dimensionality of the input while retaining physically meaningful information. The inferred low-dimensional physical parameters then serve as input to a final, compact Artificial Neural Network (ANN) trained to estimate the humidity H. This two-stage approach leverages physical knowledge for interpretability and robustness, facilitates training with smaller datasets, and results in a computationally efficient final model suitable for frugal hardware.

Section 2.1 introduces the mathematical model based on anomalous diffusion used to describe the capacitor discharge phenomenon. Section 2.2 details the Physics-Informed Dimensionality Reduction step, discussing traditional Nonlinear Least Squares (NLS), motivating the use of NPE, and describing the generation of synthetic data through GMM prior. Section 2.3 describes the final humidity estimation stage using a neural network trained on the inferred physical parameters. Finally, Section 2.4 provides a consolidated overview of the complete training and inference pipeline. We validate our method through experimental analysis which is discussed in Section 3, where we tested the accuracy of the method and also investigated the possibility of further decreasing time costs, showing that the estimation time can be effectively reduced from 4 seconds to 0.5 seconds.

1.1 Related Work

Humidity Sensors and Estimation

Estimation of air humidity is a critical task across diverse domains, from industrial processes to environmental monitoring and agriculture [1–3]. The increasing availability of low-cost humidity sensors has spurred significant interest in leveraging machine learning techniques to enhance the accuracy and reliability of humidity estimation [4,11,12]. Among the various types of humidity sensors, capacitive sensors are particularly attractive due to their favorable characteristics such as good linearity, long-term stability, and low cost [6]. These sensors operate on the principle of changes in the dielectric properties of a material in response to moisture, leading to variations in capacitance. Monitoring the electrical behavior, such as the discharge time of a capacitor, offers a direct and potentially economical route for humidity sensing [7].

Several foundational works have explored the use of machine learning to improve humidity sensing. [5] addressed the inherent inaccuracies and drift often associated with low-cost humidity and temperature sensors. This study showed the effectiveness of training various machine learning models on data from high-precision reference instruments to significantly enhance the accuracy of the low-cost sensor readings. This work highlighted the capability of machine learning to compensate for complex error sources, including non-linear sensor responses and cross-sensitivities.

A particularly relevant study in [6] presented the development of a low-cost, flexible capacitive humidity sensor designed for integration into RFID labels and microcontroller-based systems. Their approach involved screen-printing techniques to fabricate the sensor and a direct interface with a microcontroller for capacitance measurement using an RC timing method. Notably, they implemented a feedforward artificial neural network directly on the microcontroller to effectively linearize the sensor's non-linear response, achieving high accuracy. This work exemplifies the potential of combining printed electronics with embedded machine learning for accurate environmental monitoring in resource-constrained applications. The significant impact of this research is evidenced by its wide citation in subsequent works focusing on areas such as food moisture monitoring [11], flexible electronics [13], interface circuits for capacitive sensors [14], and direct sensor-to-microcontroller interfaces [15–18].

Further contributing to the development of direct sensor interfaces, in [19] the authors introduced a circuit that allows for the direct connection of capacitive humidity sensors to a microcontroller, simplifying the design and reducing cost and power consumption. Their proposed three-signal auto-calibration technique effectively compensated for offset and gain errors, demonstrating the feasibility of accurate low-cost humidity measurement systems where the sensor's inherent non-linearity becomes the primary limitation.

Beyond these initial studies, the field has seen a growing interest in employing more advanced machine learning techniques for humidity estimation. Deep learning models, particularly Recurrent Neural Networks (RNNs) [20] like LSTMs [21] and GRUs [22], have shown promise in capturing the temporal dependencies inherent in humidity data [23–25]. For instance, hybrid architectures combining Convolutional Neural Networks (CNNs) and LSTMs [21] have been explored to leverage the feature extraction capabilities of CNNs and the sequence modeling abilities of LSTMs for improved accuracy [26]. Statistical methods, including ARIMA models and Fourier analysis, also remain valuable tools for identifying patterns and forecasting humidity levels [26].

Parameter Estimation

Parameter estimation represents a fundamental problem in various scientific and engineering disciplines, aiming to determine the unknown parameters of a model based on observed data [27]. Traditional methods for parameter estimation often involve analytical techniques or iterative optimization algorithms [28,29]. For instance, when dealing with non-linear models, the method of nonlinear least squares [30] is commonly employed to find the parameters that minimize the sum of the squared differences between the observed data and the model's solution. These methods typically require a well-defined model structure and can be sensitive to initial parameter guesses and noise in the data.

In recent years, machine learning has emerged as a powerful alternative for parameter estimation, offering the ability to learn complex relationships directly from data without explicit model specification [31–33]. Neural networks [34], in particular, can be trained to map input data to the parameters of a model. When the ML model is a neural network, we refer to this task as neural parameter estimation, where the goal is to train a

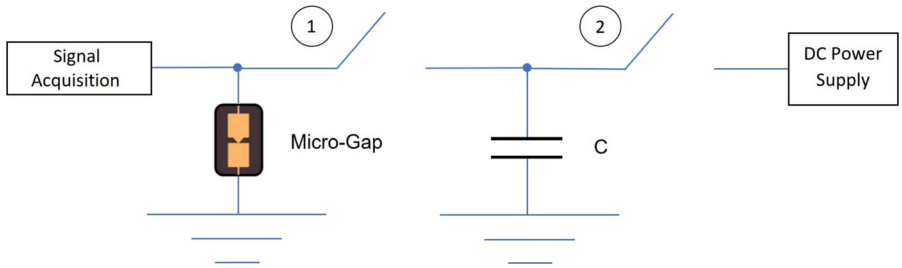

Fig. 1 Functioning Block Diagram. A capacitor is fully charged from power supply (switch 1 open, 2 closed), then it is connected to the microgap that allow electrical discharge (switch 1 closed and 2 open). Switch n.1 connects the microgap to 100 nF capacitor, which is also connected by a second one to the supply voltage. As pointed before, during the power supply charging, the capacitor is disconnected from the microgap (switch 2 closed and 1 open). By opening switch 2, the capacitor can be connected to the microgap. The voltage on the microgap is measured by a signal analyzer linked to it.

network to predict the parameters of a physical model. In our case, a neural network is trained on sensor data and corresponding physical parameters to directly estimate them. This approach can be beneficial when the underlying physical relationships are complex or not fully understood, allowing the neural network to learn the optimal parameters from the data.

Other promising machine learning methods for parameter estimation involve more advanced neural network architectures. Mixture Density Networks (MDNs) [9], for instance, can estimate the parameters of a probability distribution (e.g., means, variances, mixing coefficients) rather than just a single point value for the target parameters, allowing them to model uncertainty or multi-modal parameter solutions [10,35]. Furthermore, Physics-Informed Neural Networks (PINNs) [36] represent a significant development by integrating governing physical laws, often expressed as differential equations, directly into the network's training process. This ensures that the estimated parameters not only fit the observed data but also adhere to known physical constraints, making PINNs particularly suitable for solving inverse problems [37] common in parameter estimation across various scientific domains [38,39].

1.2 Problem Framework

Humidity Sensor The study of transport phenomena offers a robust approach for the characterization of materials. To implement novel general principles of sensing signal processing within a particular context, this research will explore the feasibility of developing an innovative humidity sensor, which is based on the discharge characteristics between microelectrodes.

The essential component of the experimental setup is the microgap, which consists of a pair of microelectrodes fabricated via photolithography, separated by a distance of a few micrometers.

The electrodes are separated by a minimum distance ranging from 1.5 to 2.5 microns. A switch is employed to connect the microgap to a 100 nF capacitor (refer to Figure 1). This capacitor is further connected via a second switch (n. 2 in Figure 1) to either a DC power supply or dedicated electronics that facilitate a 5 V voltage through a linked PC. While the capacitor is being charged by the power supply, it is disconnected from the microgap (with switch n. 2 closed and n. 1 open, as shown in Figure 1). Subsequently, when the connection between the power supply and the capacitor is disengaged, the capacitor can be connected to the microgap. The voltage across the microgap is then measured using a signal analyzer.

In scenarios where the microgap is immersed in a virtually perfect insulator or exposed to dried air, the capacitor's discharge time approaches infinity. However, in practical media, a current can traverse between the microelectrodes over finite durations. This current flow within the microgap can be utilized to characterize the ambient humidity levels as will be elucidated in subsequent sections. Furthermore, the sensor is capable of detecting relative humidity and exhibits sensitivity to temperature variations.

Humidity Estimation Framework Let $\tilde{u} \in \mathbb{R}^K$ be a time dependent observation, in our case the voltage vector over time. Let us hypothesize that there exist

a function $u : t \in \mathbb{R}^+ \mapsto u(t) \in \mathbb{R}$ such that for any $k = 0, \ldots, K-1$ holds $u_k = u(t_k)$, for a given partition $\{t_0, \ldots, t_{K-1}\}$ of a time interval $[0, \mathcal{T}]$. Since observations are made in a controlled environment with negligible perturbations, we assume that noise does not perturb this observation, and this does not lead to a loss in generality. However, it could be possible to encapsulate noise in the observations, e.g. Gaussian noise, by imposing that

$$\tilde{u}_k = u(t_k) + \epsilon_k \quad \text{for } k = 0, \ldots, K-1 \tag{1}$$

where $\epsilon_k \underset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$, with the parameter σ^2 assessing the magnitude of the noise.

The objective of this analysis is to construct a predictive model for the humidity H within the air between the two shields of the capacitor, utilizing the given discharge curve $\tilde{u}$. Humidity $H \in [0, 100]$ is defined as the proportion of water vapor present in the atmosphere. We posit the potential acquisition of an additional physical parameter, namely the environmental temperature T, which may offer statistical insights into the observed phenomenon. Consequently, our goal is to develop a machine learning architecture that accepts time series $\tilde{u} \in \mathbb{R}^K$ as input and, by conditioning on temperature, forecasts the air humidity in the sensor's surroundings. Mathematically, this problem is formulated as identifying the optimal parametric function f_θ, parameterized by the learnt parameters $\theta \in \Theta$, where Θ represents the parameter space, ensuring that $\hat{H} = f_\theta(\tilde{u}, T)$ approximates as closely as possible the true humidity value H.

Given the potential applicability of the technology to smart sensors and intelligent devices, the architecture is designed to adhere to certain epistemological constraints:

- training data is collected in labs and requires human expertise for setting the chamber, hence training with small number of samples must not influence the performance (we do not want an over-parametrized model),
- the architecture must be frugal, i.e. computationally efficient and able to run on low-powered and simple hardware.

In this paper, we propose a comprehensive solution to the aforementioned problem, which effectively satisfies the specified requirements. The approach involves projecting the time series onto a physical latent space. This is achieved by ensuring that the function $u(\cdot)$ is delineated by a physical model, such as the solution of a Partial Differential Equation (PDE), and that this model is contingent upon individual parameters that unambiguously characterize the input data $\tilde{u}$ within a low-dimensional manifold. We designate this concept as Physics-Informed Dimensionality Reduction.

2 Physics Informed Dimensionality Reduction for Humidity Estimation

In this section, we introduce and motivate our physics-aware machine learning architecture for estimating humidity using a frugal and efficient neural network. Our approach leverages physical modeling principles to improve interpretability and reliability while maintaining computational efficiency.

In Section 2.1, we present the physical model underlying the analyzed phenomenon, specifically focusing on anomalous diffusion within the capacitor microgap [40]. This model serves as the foundation for our learning framework, ensuring that the inferred parameters remain consistent with physical constraints.

Once the governing model is established, we introduce physics-informed dimensionality reduction via Neural Posterior Estimation (NPE) in Section 2.2. We begin by discussing the nonlinear least squares (NLS) method, a widely used yet computationally expensive approach for parameter estimation. We then formally introduce NPE, justifying its use from a probabilistic perspective and highlighting its advantages in terms of scalability and robustness. To further improve reliability, we employ a Gaussian Mixture Model (GMM) prior to generate an extensive synthetic dataset, allowing the NPE to generalize effectively across diverse conditions.

In Section 2.3, we describe the final stage of our pipeline: humidity estimation. Once the data is projected onto the low-dimensional physical latent space, we train a compact artificial neural network on the reduced observed dataset to accurately predict humidity levels. An illustrative scheme is shown in Figure 2.

To improve clarity and readability, we provide a structured overview of the entire methodology in Section 2.4. This section summarizes both the training routine and the real-time inference process, ensuring a coherent understanding of our approach.

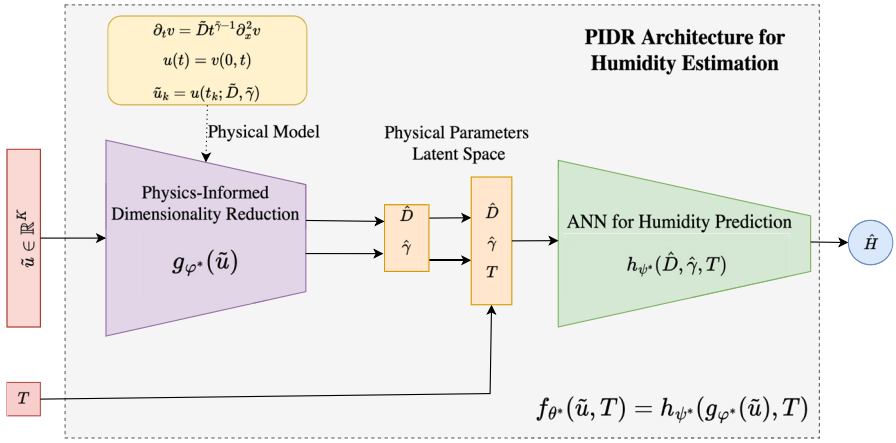

Fig. 2 Schematic representation of the model architecture. When a input signal $\tilde{u}$, it is mapped onto the low-dimensional physical latent space $\mathcal{P}$, i.e. the neural network g_{φ^*} forwards it to obtain the estimated diffusion coefficient $\hat{D}$ and the exponent $\hat{\gamma}$. Those two parameters, alongside the temperature T, initially measured by the sensor are concatenated and used as input feature for the second neural network h_{ψ^*} which outputs the prediction for the humidity level.

2.1 A Mathematical Model for Anomalous Diffusion

Let $x \in \Omega = [0, 1] \subset \mathbb{R}$ denote the spatial coordinate within the microgap, and let $t \in [0, \mathcal{T}]$ represent the temporal coordinate. The diffusion of the voltage $v(x, t)$ in the capacitor is governed by the Gaussian model with a Time-dependent Diffusion Coefficient (TdDC) [8,40], which is expressed as

$$\frac{\partial}{\partial t} v(x,t) = D t^{\gamma-1} \frac{\partial^2}{\partial x^2} v(x,t). \tag{2}$$

This equation introduces a temporal dependence in the diffusion coefficient, given by $\tilde{D}(t) = Dt^{\gamma-1}$. The parameter $\gamma \in (0, 2]$ determines the nature of the diffusion process: for $\gamma = 1$, the equation reduces to standard diffusion, whereas for $\gamma \neq 1$, it describes anomalous diffusion. In particular, when $\gamma \in (0, 1)$, the transport slows down over time, potentially leading to a complete halt (clogging). Conversely, for $\gamma \in (1, 2]$, the diffusion rate accelerates, resulting in a rapid increase in transport, which may be interpreted as a burst-like phenomenon or a micro-lightning event.

Following [8], appropriate initial and boundary conditions are imposed to model the anomalous transport dynamics within the microgap. The system is initialized with an unbalanced voltage configuration, subject to a reflecting boundary condition at $x = 0$ and an absorbing boundary condition at $x = 1$. Specifically, the initial and boundary conditions are given by

$$v(x,0) = v_0(1-x)^n, \quad \frac{\partial}{\partial x} v(0,t) = 0, \quad v(1,t) = 0, \tag{3}$$

where $v_0 > 0$ represents the initial potential applied at the left electrode, and the exponent n is set to 100 to impose a steep voltage gradient (see also [8] for further investigations).

In this framework, the function $u(t)$ describing the observed phenomenon is defined as the voltage evaluated at the left boundary,

$$u(t) = v(0,t) \quad \text{for any } t \in [0, T]. \tag{4}$$

Given a uniform temporal partition $\{t_0, \ldots, t_{K-1}\}$ of the interval $[0, \mathcal{T}]$, the observations $\tilde{u}_k$ are defined as

$$\tilde{u}_k = u(t_k) = v(0, t_k) \quad \text{for } k = 0, \ldots, K-1. \tag{5}$$

2.2 Neural Parameter Estimation with GMM Sampling for Data Augmentation

It is possible to notice that the solution of Equation (2) relies on two parameters, i.e. the diffusion coefficient D and the exponent γ. For a more compact notation, let us call those two physical parameters $p = (D, \gamma)$, and we denote by $\mathcal{P}$ the space of admissible parameters. In our case, since $D > 0$, and $\gamma \in (0, 2]$, the admissible space is $\mathcal{P} = (0, +\infty) \times (0, 2]$. In order to explicitly link the

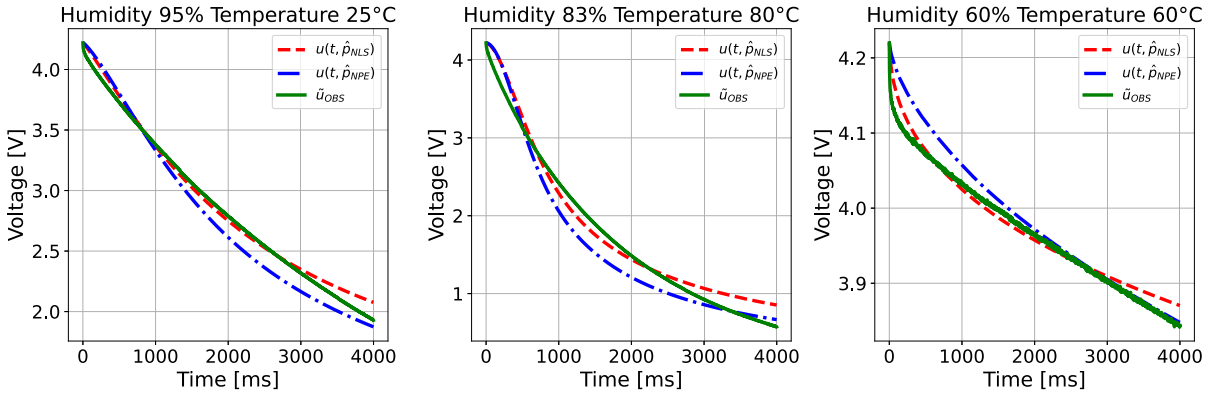

Fig. 3 Comparison of NLS and NPE in estimating model parameters based on observed data (green curve) across different humidity and temperature conditions. The black curves represent physical model solution using parameters estimated via NPE, while the red curves correspond to those estimated with NLS, respectively $u(t; \hat{p}_{\text{NPE}})$ and $u(t; \hat{p}_{\text{NLS}})$. The voltage, expressed in volts, is plotted over a 4000-millisecond observation window.

solution of the model to the physical parameters, we write the physical model as $u(t; p)$.

The core idea behind our method is to enhance computational efficiency by reducing the input space dimension (K) to a low-dimensional space that maximizes the significance of the new features. This latent space corresponds to the space of the physical parameters, i.e., $\mathcal{P}$, which is a low-dimensional space. In our case, $|\mathcal{P}| = 2$, meaning $|\mathcal{P}| \ll K$.

In this section, we focus on the *inverse problem* [27] of finding the parameters $\tilde{p} \in \mathcal{P}$ of the model that generate the observed sample $\tilde{u}_k = u(t_k; \tilde{p})$ for a partition of $[0, \mathcal{T}]$, indexed by $k = 0, \ldots, K-1$. The state-of-the-art technique for parameter estimation is the *Non-Linear Least Squares* (NLS) method [30], [41]. In our experimental evaluations, we used the Bayesian implementation of `lsqcurvefit` [42] in MATLAB. Illustrative comparisons between model solutions obtained from parameters estimated both with NLS and NPE are shown in Figure 3.

Non-linear Least Square Method The NLS method aims to minimize the discrepancy between the observed data and the model predictions by solving the least squares problem. Specifically, for a fixed observation $\tilde{u}$, the routine finds the optimal physical parameters by solving

$$\hat{p} \in \arg\min_{p \in \mathcal{P}} \sum_{k=0}^{K-1} |u(t_k; p) - \tilde{u}_k|^2. \tag{6}$$

That is, the optimal parameters $\hat{p}$ are those that minimize the ℓ^2-norm between the observed sample $\tilde{u} \in \mathbb{R}^K$ and the solution of the TdDC model with parameters $p \in \mathcal{P}$. The most commonly used approach to solve this problem is the Levenberg-Marquardt algorithm [42], which interpolates between the Gauss-Newton method and gradient descent. This is achieved by iteratively updating the parameters via

$$p^{(i+1)} = p^{(i)} - (J^T J + \lambda I)^{-1} J^T r, \tag{7}$$

where J is the Jacobian of the residuals, $r = u(t_k; p) - \tilde{u}_k$ is the residual vector, and λ is the damping parameter that dynamically adjusts the balance between Gauss-Newton and gradient descent steps.

Although this method provides robust and precise estimates, its computational cost makes it impractical for real-time applications and resource-constrained devices such as smart sensors. The primary limitation is that each iteration requires solving the PDE for the current parameters, which is computationally expensive. Since multiple evaluations are needed until convergence, even highly optimized software can lead to excessive time delays and computational burden.

Nevertheless, due to its high precision and robustness, we used this technique as a preliminary step to determine the *ground-truth* physical parameters of the training set, which are otherwise unknown. In the following section, we discuss our proposed method which mitigates these computational challenges.

Neural Parameter Estimation Neural Parameter Estimation (NPE) refers to the use of neural networks to infer the parameters of a physical model from observed data. Unlike traditional optimization techniques such as Non-Linear Least Squares or Bayesian inference, NPE leverages the approximation capabilities of neural networks to learn complex mappings between input observations and the underlying parameters of the system. Formally, NPE defines a function, parameterized by a neural network, that maps observed data to the physical parameter space. Given a neural network with trainable weights $\varphi \in \Phi$, the estimation function is expressed as

$$g_\varphi : u \in \mathbb{R}^K \longmapsto g_\varphi(u) \in \mathcal{P}, \tag{8}$$

where $g_\varphi(u)$ represents the predicted physical parameters associated with the observation u. The parameters φ are learned by minimizing a suitable loss function over a training set.

Let $\mathcal{D} = \{(u^n, p^n) \mid n = 1, \ldots, N\} \subset \mathbb{R}^K \times \mathcal{P}$ be a dataset containing observations $u^n \in \mathbb{R}^K$ generated from the physical model with corresponding parameters $p^n \in \mathcal{P}$, where each component satisfies $u_k^n = u(t_k; p^n)$. To formally describe the estimation problem, we introduce a probabilistic framework, following [9]. Specifically, we assume that u and p are absolutely continuous random variables with a joint probability density function $\pi(u, p)$ satisfying

$$\int_{\mathbb{R}^K \times \mathcal{P}} \pi(u, p)\, dp\, du = 1. \tag{9}$$

By applying the standard decomposition of joint probability distributions, we express $\pi(u, p)$ as

$$\pi(u, p) = \pi(p|u)\pi(u), \tag{10}$$

where $\pi(p|u)$ represents the posterior distribution of the parameters given the observation u, and $\pi(u)$ denotes the prior distribution of the observations. Since $\pi(u)$ is typically high-dimensional and difficult to model explicitly, NPE focuses on learning an approximation of the posterior distribution $\pi(p|u)$. In this framework, parameter estimation is formulated as computing the expected value of p given an observation u, i.e. $\mathbb{E}[p|u]$. This expectation serves as the optimal estimator in the mean-squared error sense and provides a probabilistic interpretation of the inferred parameters.

Learning the Map Learning the NPE map $g_\varphi(u)$ consists of finding the optimal weight configuration φ^* that minimizes the empirical error over a given training set $(u^n, p^n)_{n=1}^N$. This corresponds to solving the optimization problem:

$$\varphi^* \in \arg\min_{\varphi \in \Phi} \frac{1}{2N} \sum_{n=1}^{N} \| g_\varphi(u^n) - p^n \|_2^2, \tag{11}$$

where $\| \cdot \|_2$ denotes the ℓ^2-norm. This formulation represents a standard regression problem, where the objective is to minimize the average discrepancy between the observed responses and the predicted outputs. The function being minimized is commonly referred to as the *mean squared error* (MSE) loss, which can be optimized using various stochastic algorithms, such as Stochastic Gradient Descent or Adam [43].

Denoting the empirical loss function as $\mathcal{L}_N(\varphi)$, the optimal solution φ^* minimizes $\mathcal{L}_N(\varphi)$, making g_{φ^*} the best possible approximation given N training samples. As N increases, the model's predictions become more accurate. This intuition is further supported by the following theoretical result, which connects neural network regression with probabilistic parameter estimation [34,44] and, more broadly, with generative modeling frameworks [45].

As $N \to \infty$, the empirical loss function converges to its expected counterpart due to the Central Limit Theorem:

$$\begin{aligned} \lim_{N\to\infty} \mathcal{L}_N(\varphi) &= \lim_{N\to\infty} \frac{1}{2N} \sum_{n=1}^{N} \| g_\varphi(u^n) - p^n \|_2^2 \\ &= \frac{1}{2} \mathbb{E}(\| g_\varphi(u) - p \|_2^2) = \mathcal{L}(\varphi), \end{aligned} \tag{12}$$

where the expectation is taken with respect to the joint probability distribution $\pi(p, u)$. Exploiting the linearity of expectation, we can express the loss function as

$$\begin{aligned} \mathcal{L}(\varphi) &= \frac{1}{2} \sum_{q=1}^{Q} \mathbb{E}(g_\varphi(u)_q - p_q)^2 \\ &= \frac{1}{2} \sum_{q=1}^{Q} \int_{\mathbb{R}^K \times \mathcal{P}} (g_\varphi(u)_q - p_q)^2 \pi(u, p)\, dp\, du, \end{aligned} \tag{13}$$

where p_q denotes the q-th component of the physical parameter vector p. To determine the optimal φ^* minimizing $\mathcal{L}(\varphi)$, we impose the optimality condition

$$\frac{\delta \mathcal{L}(\varphi^*)}{\delta g_\varphi(u)_q} = 0, \quad \forall q = 1, \ldots, Q. \tag{14}$$

Computing the variational derivative and applying this condition yields

$$\int_{\mathbb{R}^K \times \mathcal{P}} (g_{\varphi^*}(u)_q - p_q)\pi(p, u)\, dp\, du = 0. \tag{15}$$

Applying the decomposition $\pi(p, u) = \pi(p|u)\pi(u)$ and integrating with respect to $\pi(p|u)dp$ we obtain

$$\begin{aligned} &\int_{\mathbb{R}^K \times \mathcal{P}} (g_{\varphi^*}(u)_q - p_q)\pi(p|u)\pi(u)\, dp\, du = 0 \\ &\iff \int_{\mathbb{R}^K} [\mathbb{E}(g_{\varphi^*}(u)_q|u) - \mathbb{E}(p_q|u)]\pi(u)\, du = 0 \end{aligned} \tag{16}$$

which is true if and only if $\mathbb{E}(g_{\varphi^*}(u)_q|u) = \mathbb{E}(p_q|u)$, and since $\mathbb{E}(g_{\varphi^*}(u)_q|u) = g_{\varphi^*}(u)_q$. This sums up to the condition

$$g_{\varphi^*}(u)_q = \mathbb{E}(p_q|u). \tag{17}$$

Equation (17) establishes a fundamental result: training a neural network for regression via MSE minimization is equivalent to computing the conditional expectation of the physical parameters given the observed data, without requiring explicit assumptions about the distributions of u and p.

GMM Sampling To train the NPE model, we require a large synthetic dataset, denoted as

$$\mathcal{D}_{\text{train}} = \{(u^n, p^n) : n = 1, \ldots, N\}.$$

In our experimental setting, however, we have access to only a limited number of observed curves, for which the exact physical parameters remain unknown. Specifically, the available dataset

$$\{\tilde{u}^n : n = 1, \ldots, N_{\text{obs}}\}$$

is small, with N_{obs} around 100. To ensure robust training of the NPE map g_{φ^*}, a substantially larger number of training samples is needed–ideally, $N \to \infty$. Consequently, we must augment the data.

When performing data augmentation and synthetic data generation, selecting an appropriate prior distribution $\pi(p)$ over the parameter space $\mathcal{P}$ is crucial. An unrealistic or mis-specified prior can severely degrade the quality of parameter estimation.

Given a prior distribution $\pi(p)$, the data generation process follows a straightforward two-step procedure:

1. Sample $p^n \sim \pi(p)$ for $n = 1, \ldots, N$.
2. Numerically solve the PDE to obtain $u_k^n = u(t_k, p^n)$ at each time step t_k.

This results in a training dataset

$$\mathcal{D}_{\text{train}} = \{(u^n, p^n) : n = 1, \ldots, N\},$$

where each sample pair $(u^n, p^n) \sim \pi(u, p)$.

A key challenge is the selection of a suitable prior $\pi(p)$. A poorly chosen prior can lead to unreliable parameter estimation. For instance, a uniform distribution over $\mathcal{P}$ performed poorly, as it failed to capture correlations among parameters. To address this point, we leveraged the robustness of the Nonlinear Least Squares (NLS) method to estimate the generating parameters of the observed curves $\tilde{u}$. Denoting these estimates as $\hat{p}_{\text{NLS}}$, we assessed their quality by computing the residual norm for each observation:

$$\sum_{k=1}^{K} |\tilde{u}_k - u(t_k; \hat{p}_{\text{NLS}})|^2.$$

This allowed us to construct an empirical dataset

$$\mathcal{D}_{\text{obs}} = \{(\tilde{u}^n, \hat{p}^n_{\text{NLS}}) : n = 1, \ldots, N_{\text{obs}}\},$$

where we treat the NLS-estimated physical parameters as approximate ground truth.

Since $\mathcal{P}$ is a continuous space, a more flexible prior was necessary. Instead of assuming a uniform distribution, we modeled $\pi(p)$ as a mixture of Gaussian distributions:

$$\pi(p) = \sum_{j=1}^{J} \alpha_j \mathcal{N}(\mu_j, \Sigma_j),$$

where α_j are the mixture weights, and $\mathcal{N}(\mu_j, \Sigma_j)$ denotes a Q-dimensional Gaussian with mean $\mu_j \in \mathbb{R}^Q$ and covariance matrix $\Sigma_j \in \mathbb{R}^{Q \times Q}$. This choice

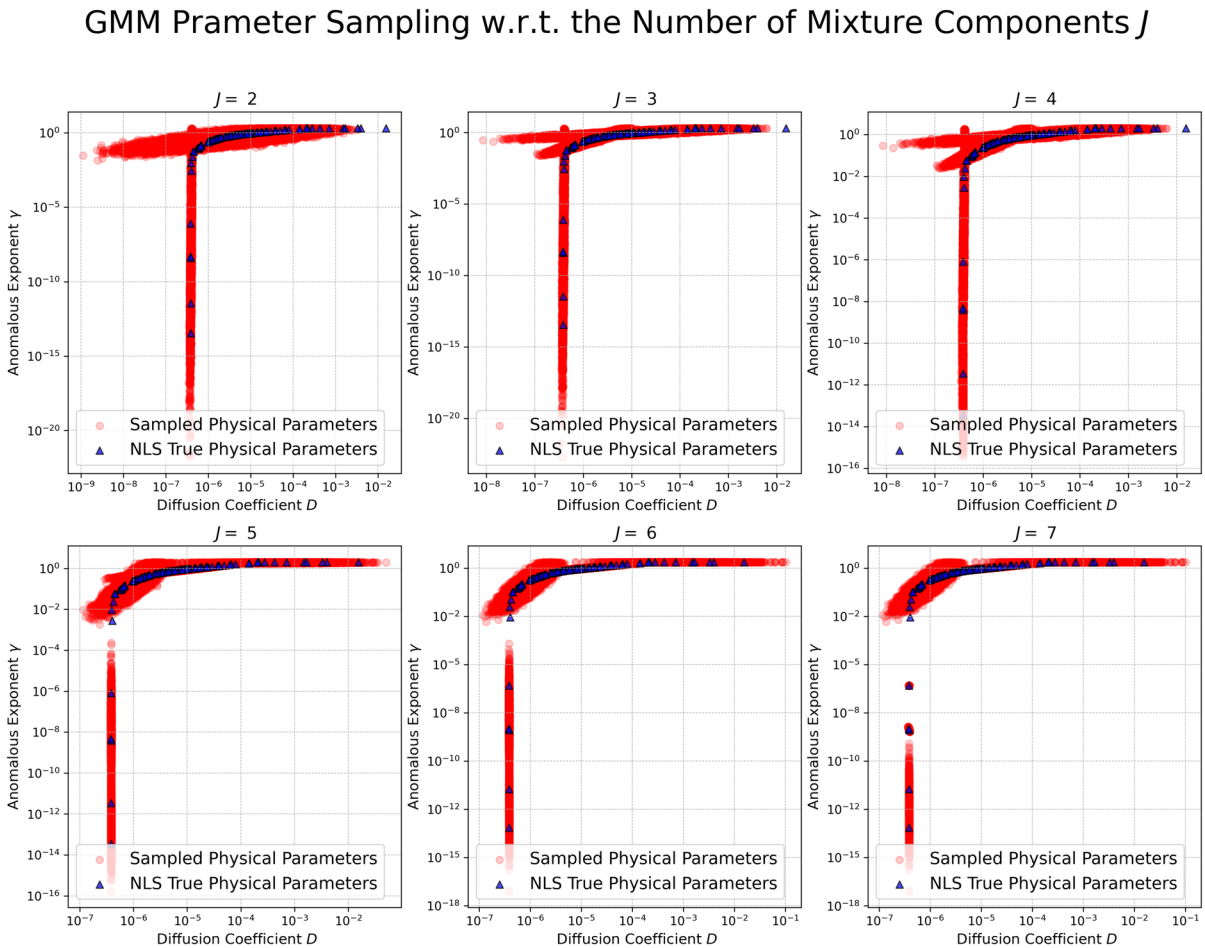

Fig. 4 GMM sampling with respect to the number of mixture components $J \in \{2, 3, \ldots, 9\}$. The scatter plots, with both axes logarithmically scaled, depict the coefficients D and γ estimated using NLS on the observed curves, represented as black triangles, which are considered as the ground truth, and the red circles illustrate the 100,000 samples derived from the various prior distributions. It is observed that as the number of mixture components increases, the sampling algorithms have a tendency to overfit certain modes, particularly those associated with a lower value of γ. Conversely, when employing a smaller value for J, some regions become under-represented, such as those exhibiting large values of D. Therefore, a suitable balance between model complexity, as well as the exploration and exploitation of data samples, was identified in $J = 5$, which was subsequently utilized to sample the dataset for training the NPE. The selection of $J = 5$ was undertaken in order to minimize the Wasserstein-2 distance [46] between the empirical data and the estimated GMM, see Table 1.

allowed for a more accurate representation of the underlying data structure, leading to a more effective synthetic dataset and, ultimately, improved performance of the NPE model. In Figure 4, we show the sampled parameters for different number of mixture components J. Algorithm 1 summarizes the procedure followed to sample the synthetic dataset, used to train the NPE.

Training the Parameter Estimation Training the NPE on the simulated dataset $\mathcal{D}_{\text{train}}$ allows learning the map g_{φ^*}, where φ^* represents the optimal weight vector of the neural network that minimizes the empirical loss function $\mathcal{L}_N(\varphi)$. Given that N is large (in our experiments, approximately 10^5 samples), the trained network provides an accurate and precise estimate of the conditional expected value of the parameters given the observations. This ensures that the learned posterior distribution aligns closely with the underlying data-generating process.

Algorithm 1 Synthetic Data Augmentation with GMM Sampling

Require: Lab measurements $\tilde{u}^n \in \mathbb{R}^K$ for $n = 1, \dots, N_{\text{obs}}$
Ensure: Sampled dataset $\mathcal{D}_{\text{train}} = \{(u^n, p^n) \mid n = 1, \dots, N\}$
1: **for** $n = 1, \dots, N_{\text{obs}}$ **do**
2: Compute estimated parameters: $\hat{p}^n_{\text{NLS}} \leftarrow \text{NLS}(\tilde{u}^n)$
3: **end for**
4: $\mathcal{D}_{\text{obs}} \leftarrow \{(\tilde{u}^n, \hat{p}^n_{\text{NLS}}) \mid n = 1, \dots, N_{\text{obs}}\}$
5: Estimate GMM parameters $\{\hat{\alpha}_j, \hat{\mu}_j, \hat{\Sigma}_j\}_{j=1}^J$ from $\{\hat{p}^n_{\text{NLS}} \mid n = 1, \dots, N_{\text{obs}}\}$
6: Fix N very large and sample $p^n \sim \pi(p) = \sum_{j=1}^J \hat{\alpha}_j \mathcal{N}(\hat{\mu}_j, \hat{\Sigma}_j)$
7: **for** $n = 1, \dots, N$ **do**
8: Numerically solve the PDE model: $u^n_k = u(t_k; p^n)$ for any $k = 0, \dots, K-1$
9: **end for**
10: Return $\mathcal{D}_{\text{train}} \leftarrow \{(u^n, p^n) \mid n = 1, \dots, N\}$

In our application, when the sensor acquires a new time series $\tilde{u}$, the trained network g_{φ^*} maps the signal onto a two-dimensional latent space representing the physical parameters. Specifically, denoting the predicted diffusion coefficient by $\hat{D}$ and the predicted exponent by $\hat{\gamma}$, we obtain

$$\begin{cases} \hat{D} &= g_{\varphi^*}(\tilde{u})_1 = \mathbb{E}[D|\tilde{u}] \\ \hat{\gamma} &= g_{\varphi^*}(\tilde{u})_2 = \mathbb{E}[\gamma|\tilde{u}]. \end{cases} \tag{18}$$

Thus, the first output neuron of the NPE network corresponds to the estimated diffusion coefficient, while the second neuron provides the predicted exponent.

To optimize the network, we employ the ADAM optimizer [43]. The training objective is to minimize the MSE loss, ensuring that the estimated posterior remains close to the true parameter distribution.

2.3 Humidity Estimation with Artificial Neural Networks

Now that we have accurately trained the Neural Posterior Estimator (NPE), the map g_{φ^*} serves as a nonlinear function that projects the input time series onto a low-dimensional, interpretable physical space. This transformation effectively acts as a denoiser, as it compresses high-dimensional data into a two-dimensional representation while maximizing expressiveness.

The final block of the architecture consists of a shallow regression network designed to predict humidity based on three physical inputs: the physical parameters inferred by the NPE and the temperature of the chamber where the capacitor is placed. Specifically, we define the regression network as a mapping

$$\begin{aligned} h_\psi : \mathcal{P} \times \mathbb{R} &\longrightarrow \mathbb{R} \\ (\hat{D}, \hat{\gamma}, T) &\longmapsto \hat{H} = h_\psi(\hat{D}, \hat{\gamma}, T), \end{aligned} \tag{19}$$

where $\hat{D}$ and $\hat{\gamma}$ are the predicted physical parameters from the NPE, T is the temperature, and $\hat{H}$ is the predicted humidity.

The optimal weights ψ^* are obtained by minimizing the Mean Squared Error (MSE) loss function with the ADAM optimizer [43], solving the optimization problem

$$\psi^* \in \arg\min_{\psi \in \Psi} \frac{1}{2N_{\text{obs}}} \sum_{n=1}^{N_{\text{obs}}} (h_\psi(\hat{D}^n, \hat{\gamma}^n, T^n) - H^n)^2. \tag{20}$$

Since the humidity prediction model is trained on real laboratory measurements, the regression network must remain shallow. This is because temperature and humidity are not directly linked to the discharge curve through an explicit physical model. Instead, their relationship is a black-box function that we learn from data using the overall architecture.

To train the ANN for humidity estimation, we use observed experimental data consisting of time series measurements $\tilde{u}^n$ along with the corresponding temperature T^n and humidity H^n (i.e., the target variable) for $n = 1, \dots, N_{\text{obs}}$. Since the model must be autonomous and efficient, while also leveraging the feature extraction capabilities of the trained NPE g_{φ^*}, we train h_ψ on the dataset

$$\mathcal{D} = \{(g_{\varphi^*}(\tilde{u}^n), T^n, H^n) \mid n = 1, \dots, N_{\text{obs}}\}, \tag{21}$$

where $g_{\varphi^*}(\tilde{u}^n) = (\hat{D}, \hat{\gamma})$.

Thus, training the overall model consists of two phases:

1. Training the NPE g_φ on realistic synthetic data to learn the mapping from time series to physical parameters.
2. Training the final regression layer h_ψ on observed experimental data, where input features are first dimensionally reduced using the trained NPE g_{φ^*}.

Algorithm 2 Training the Model

Require: Lab measurements $\tilde{u}^n \in \mathbb{R}^K$, with temperature T^n and humidity H^n for $n = 1, \ldots, N_{\text{obs}}$

Ensure: Trained blocks g_{φ^*} and h_{ψ^*}

1: Use Alg. 1 to simulate the dataset $\{(u^n, p^n) \mid n = 1, \ldots, N\}$ with GMM to train the NPE for dimensionality reduction g_φ

2: Train g_φ by minimizing the empirical loss $\mathcal{L}_N^g(\varphi) = \frac{1}{2N} \sum_{n=1}^{N} \|g_\varphi(u^n) - p^n\|_2^2$ with stochastic optimizer, obtaining the weights φ^*

3: Train the humidity predictor h_ψ by mapping the observed data onto the latent space, jointly with the temperature, i.e. on $\{(g_{\varphi^*}(\tilde{u}^n), T^n, H^n) \mid n = 1, \ldots, N_{\text{obs}}\}$ by minimizing the empirical loss $\mathcal{L}_{N_{\text{obs}}}^h(\psi) = \frac{1}{2N_{\text{obs}}} \sum_{n=1}^{N_{\text{obs}}} (h_\psi(g_{\varphi^*}(\tilde{u}^n), T^n) - H^n)^2$

2.4 The Algorithm

In this section, we provide a summary and some remarks on the overall algorithm. Our objective is to learn a parametric function f_θ that maps a potential discharge observation and the corresponding ambient temperature to an estimate of humidity. To achieve this, we employ a two-block architecture, illustrated in Figure 2, which leverages the underlying physical model to construct an efficient humidity estimator.

The first block, denoted as g_{φ^*}, performs physics-informed dimensionality reduction (NPE), projecting the time series data onto a two-dimensional latent space. The second block, a shallow neural network h_{ψ^*}, takes as input the extracted physical parameters along with the chamber temperature and predicts the corresponding humidity value.

Formally, we express this relationship as:

$$f_{\theta^*}(\tilde{u}, T) = h_{\psi^*}(g_{\varphi^*}(\tilde{u}), T) = \hat{H}, \tag{22}$$

where $\hat{H}$ represents the predicted humidity. The parameter space $\Theta = \Phi \times \Psi$ is the Cartesian product of the parameter spaces of the two blocks. Consequently, the optimal parameters for the architecture are given by $\theta^* = (\varphi^*, \psi^*)$, obtained through a two-stage optimization procedure. The NPE is trained on a large-scale synthetic dataset, while the final regression network is trained on the collected experimental data.

Algorithm 2 outlines the training routine employed to optimize the architecture.

3 Experiments and Results

This section presents the experiments conducted to evaluate our approach for fast humidity estimation using Physics-Informed Dimensionality Reduction (PIDR). Section 3.1 describes the experimental setup, including the selected architectures, hyperparameters, and dataset construction and split. Section 3.2 analyzes the results in terms of accuracy and computational efficiency, comparing the performance of the architecture when employing the NLS method versus NPE for parameter estimation. The results indicate that applying PIDR significantly improves model predictions–without it, achieving accurate predictions is not guaranteed. Crucially, using NPE instead of NLS for PIDR substantially reduces both computational overhead and required resources, decreasing evaluation time from seconds to milliseconds. This improvement highlights the potential for real-world deployment in sensor and IoT devices.

3.1 Implementation Details

This section outlines the implementation details necessary to assess our method. We first describe the dataset used for training, including the laboratory setup for data collection. We then provide an overview of the chosen architectures, training setup, and computational resources.

Settings and data collection Experiments were executed utilizing a climatic chamber, model LC/64/70/3. The selection of specific temperature and humidity setpoints was deliberately chosen to align with the sensor's primary intended application, while also providing a broader characterization within the technical constraints of the experimental setup. The main focus was on conditions relevant to proton-exchange membrane (PEM) fuel cells, which typically operate at elevated temperatures (e.g., above 60 °C) and high relative humidity (60% to 90% RH) to ensure optimal performance. While these demanding conditions were prioritized, the experimental scope was extended to a wider temperature range (from 25 °C to 80 °C) for a more comprehensive validation. It is also important to note that the choice of discrete humidity values was dictated by the climatic chamber's technical specifications, which support a minimum resolution of 1% RH.

List of Temperatures (°C)																		
25			40			50			60			70			80			
List of Relative Humidities (%)																		
50	53	55	58	60	63	65	68	70	73	75	78	80	83	85	88	90	93	95

Fig. 5 Label of testing temperature and relative humidity values. Tests were performed by setting six temperature values 19 relative humidity values for each of them, requiring 1 hour to stabilize every selected humidity inside the chamber before initiate the signal acquisition.

Within this system, it is feasible to configure three distinct parameters:

- Temperature (T)
- Relative Humidity (Rh)
- Stabilization time

The following label (Figure 5) shows the temperature and relative humidity values set in the performed tests.

In these experiments, the microgap is interfaced with dedicated electronic equipment which facilitates the charging and discharging processes on the sample and records the output voltage values through the use of specific freeware (YAT 2.1.0). For each selected temperature, measurements were conducted by varying the relative humidity of the climatic chamber. Prior to the initiation of data acquisition, an approximate duration of 60 minutes is allowed for the humidity within the chamber to stabilize.

The data recording span is 4 seconds, with a sampling frequency of 1 sample/ms, thus each acquisition yields 4000 output voltage values. Given the influence of both temperature and humidity on the measurement, various discharge dynamics in the output values can be observed, with elevated temperatures and humidity levels resulting in higher discharge rates.

Figure 6 presents illustrative data series highlighting the relationship between capacitor discharge, temperature (measured in degrees Celsius), and humidity (represented in percentiles). Reduced temporal observation interval, e.g., 100 milliseconds (see Figure 7), are considered to evaluate the minimum time necessary to distinguish between discharge curves corresponding to distinct temperature and humidity settings.

Dataset We constructed our dataset using three attributes: the voltage discharge $\tilde{u} \in \mathbb{R}^K$, the corresponding temperature T, and the humidity percentage H (the target variable). Our observed dataset, $\mathcal{D}\text{obs} = \{(\tilde{u}^n, T^n, H^n) \mid n = 1, \ldots, N_{\text{obs}}\}$, consists of $N_{\text{obs}} = 118$ observations collected in a controlled environment. Humidity levels were systematically increased from 50% to 53%, 55%, 58%, and so on up to 98%. Temperatures were recorded at 25°C, 40°C, 50°C, 60°C, 70°C, and 80°C. Due to the limited number of observations, we performed a 95%-5% split on the observed data. Results were averaged over 5 different random seeds to ensure reproducibility. This choice reflects a realistic scenario in which data collection is expensive, and we aim to maximize the available information for the training process. The portion of the data used for training was also used for sampling and augmenting the dataset for parameter estimation (as described in Algorithm 1). For the simulated data used to train the NPE, we generated a large number of samples (100,000), then filtered out physical parameters outside the admissible set $\mathcal{P}$, resulting in a final set of 95,758 simulated samples.

Architectures and Hyper-parameters Since our goal is to develop a model suitable for deployment on resource-constrained hardware, we designed a compact architecture consisting of two main components: the NPE network for PIDR, g_φ, which maps the input time series onto a two-dimensional physical space, and the humidity prediction network, h_ψ, which maps the triplet (D, γ, T) to the predicted humidity value.

The NPE network g_φ is a simple multilayer perceptron (MLP) [47] with Rectified Linear Unit (ReLU) activations [48]. It consists of two layers, with 2048 neurons in the first and 512 in the second. The model was trained for 1000 epochs on the GMM-simulated dataset with $J = 5$ mixture components, using a batch size of 256. Training was performed with the ADAM optimizer [43], combined with cosine annealing [49] for learning rate scheduling, with a maximum of 20 annealing rounds. The number of mixture components was set to $J = 5$ as it minimizes the Wasserstein-2 distance [46] between the empirical distribution of the (D, γ) parameters and the estimated GMM prior (see Table 1). This choice avoids overfitting the observed data while promoting a more diverse and robust train-

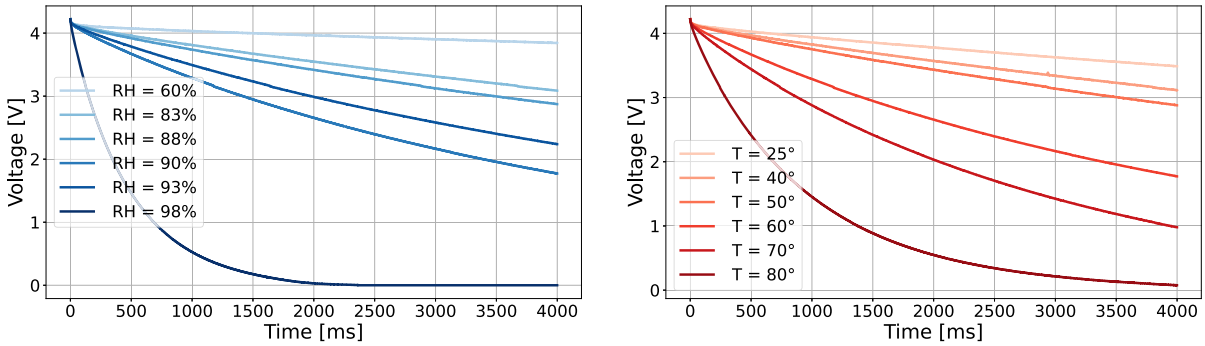

Fig. 6 Voltage discharge measurements over 4 seconds, obtained for (left) different percentage of relative humidity, at fixed temperature $T = 60°C$: (right) different temperature values, at fixed relative humidity $H = 90\%$. Higher humidity and temperature values correspond to faster discharge rates.

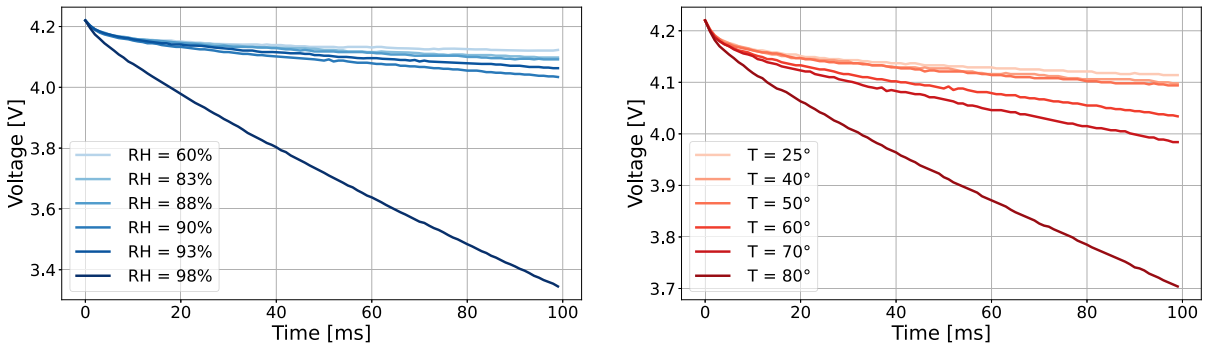

Fig. 7 Voltage discharge measurements over 100 milliseconds, obtained for (left) different percentage of relative humidity, at fixed temperature $T = 60°C$: (right) different temperature values, at fixed relative humidity $H = 90\%$. Higher humidity and temperature values correspond to faster discharge rates. Working at shorter time intervals allows us to investigate the response of the sensor to rapid environmental changes, as well as the sensitivity of the device.

Table 1 Wasserstein-2 distance [46] between the empirical distribution of the (D, γ) parameters and the estimated GMM prior as a function of the number of mixture components J. The optimal value of J corresponds to the lowest Wasserstein distance, indicating better coverage of the parameter space without mode collapse.

Mixture Components J	1	2	3	4	**5**	6	7
Wasserstein-2	1.8076	0.6898	0.7256	0.4779	**0.4082**	1.2239	0.7863

ing set, as evident from Figure 4. The initial learning rate was set to 0.001. The input time series was rescaled to [0, 1], and target parameters were log-transformed (base 10) to enhance differentiation, as diffusion coefficients vary within a narrow range.

The final mapping function, h_ψ, for humidity prediction is implemented as a single-layer shallow neural network with a ReLU activation function. As in previous steps, the input physical parameters are log-transformed and standardized, along with the output variable. To ensure the predicted humidity remains within the valid range, we apply a cut-off function that maps the unstandardized output to the closest value in the training set. The number of neurons is tuned for each experiment within the range $\{5, 10, \ldots, 220\}$, with the best-performing configuration typically around 80 neu-

Table 2 Performance summary highlighting the advantages of the PIDR method in terms of RMSE across different test set splits. We consider RMSE below 3% as a good result, given the training dataset's sensitivity of 3%. The models are trained on the voltage discharge with time duration $\mathcal{T} = 4\,s$. The table compares parameter estimation using NLS and NPE, noting that PIDR with NLS provides an upper bound for PIDR with NPE but requires significantly more computation time. Dimensionality reduction is essential, as the model without it yields errors exceeding 10, making it impractical. Applying PIDR allows to achieve RMSE below the sensitivity threshold, with errors below 2% for humidity values under 60% and above 75%.

		Model		
		PIDR (NLS) + ANN	PIDR (NPE) + ANN	ANN w/o PIDR
Test Data Split	All	2.236 ±0.957	2.463 ±0.744	13.759 ±1.806
	H below 60%	1.816 ±1.486	1.816 ±1.486	17.334 ±2.033
	H above 60%	2.247 ±0.857	2.572 ±0.667	12.194 ±1.826
	H below 75%	2.370 ±1.243	2.453 ±0.929	12.907 ±1.228
	H above 75%	1.897 ±0.766	2.338 ±0.765	15.194 ±2.942

rons. The network is trained using the ADAM optimizer for a maximum of 2000 epochs with initial learning rate of 0.001. Since the model is trained directly on the observed dataset, we do not use batch-based training, as the dataset is limited.

Finally, in the experiments reported in Table 2, when PIDR is not applied (third column, "ANN w/o PIDR"), we fine-tune a single-layer MLP that directly maps the observations, along with the temperature, to the humidity value. The number of neurons is tuned within the range $\{5, 10, \ldots, 500\}$. A larger model is not used in this case, as training is performed directly on the small observed dataset rather than a large simulated one. The network is trained using the ADAM optimizer for 2000 epochs with a learning rate of 0.001.

Software and Computational Resources The PDE solutions are numerically computed in MATLAB using `pdepde` function. The NLS parameter estimation is implemented using the `lsqcurvefit` function from the `Optimization Toolbox`. These computations are executed locally on an Apple M1 processor. The remaining code is implemented in Python. In particular, the NPE training is performed on GPUs–specifically, NVIDIA RTX 8000–using the high-performance computing facilities at Politecnico di Torino with the `PyTorch` library [50]. However, the trained NPE model is evaluated on CPUs to simulate deployment on resource-constrained sensor hardware. The humidity prediction model is trained locally on the Apple M1 processor using the `scikit-learn` library [51].

3.2 Results Analysis

Evaluation Metric To assess the predictive performance of our approach, we use the Root Mean Squared Error (RMSE) [52], computed on the test set and averaged across five different train-test splits with varying random seeds. Given predicted humidity values $\hat{H}^n$ and corresponding ground truth values H^n for $n = 1, \ldots, N_{\text{test}}$, the RMSE is defined as

$$\text{RMSE} = \sqrt{\frac{1}{N_{\text{test}}} \sum_{n=1}^{N_{\text{test}}} (\hat{H}^n - H^n)^2} \tag{23}$$

RMSE is preferred over Mean Squared Error (MSE) as it provides an interpretable measure of the average prediction error in the same units as the target variable. Given that the humidity levels in our observed dataset are spaced by 3% and 2%, we consider an RMSE below 3% indicative of a well-performing model, while an RMSE below 2% signifies highly accurate predictions.

As detailed in the following sections, the proposed PIDR methodology consistently achieves RMSE values below 3%, whether employing NLS or NPE for parameter estimation–where NLS serves as an upper bound reference for our approach. Notably, our results demonstrate that NPE can even surpass this upper bound by reducing the temporal duration of the input signal, achieving an RMSE below 2%, as reported in Table 4.

Table 3 Relevance of physical features in humidity prediction: a comparison between NLS and NPE. The reported metric is test RMSE on humidity. The table examines how different features affect the model's predictive performance when estimating parameters using NLS and NPE. To enhance data separation and maximize information, we applied a logarithmic transformation to the D and γ coefficients. The best performance is obtained when including both model parameters and temperature. However, the feature pair (D, T) achieves similar performance, suggesting that the anomalous exponent has a limited impact on prediction–except in cases where humidity is low and measured at low temperatures. Rows where the test RMSE is below 3% are highlighted in grey. The results also emphasize the importance of conditioning the model with the temperature feature, which improves robustness and accuracy in humidity estimation.

Physical Features			Parameter Estimation Method	
D	γ	T	NLS	NPE
✓	✓	✓	2.236 ±0.957	2.463 ±0.744
✓	✓	✗	9.979 ±1.268	9.814 ±1.419
✓	✗	✓	2.366 ±0.852	2.517 ±0.512
✓	✗	✗	9.918 ±1.746	9.944 ±1.359
✗	✓	✓	13.996 ±2.431	12.780 ±2.102
✗	✓	✗	10.005 ±1.921	12.934 ±1.683
✗	✗	✓	14.029 ±2.441	14.029 ±1.992

Advantages of PIDR

In this section, we discuss the advantages of applying PIDR to the humidity prediction task by comparing three different models: two variations of the PIDR architecture–one using NLS for parameter estimation, which is computationally more expensive, and the other using NPE, which is more efficient–against an ANN trained directly on the observed data. The ANN model maps the discharge curve, coupled with temperature, directly to the humidity value.

We evaluate the predictive performance of these models using the test RMSE, as reported in Table 2, where all curves are observed over a 4-second time interval. The results clearly indicate that the model without PIDR fails to learn the mapping due to the small sample size. In contrast, both versions of PIDR (with NLS and NPE) achieve test RMSE values consistently below the 3% threshold, whereas the ANN without dimensionality reduction model remains above 10%, far from an acceptable performance level.

Further insights can be gained by examining test RMSE across different humidity ranges (Table 2). Notably, the PIDR architecture maintains strong performance across all ranges. In particular, when the ground truth humidity is below 60%, both PIDR models achieve an RMSE of 1.816, demonstrating high reliability in this challenging region, where curve variations are nearly imperceptible. Additionally, the model–especially with NLS parameter estimation–achieves highly accurate predictions for humidity values above 75%.

These results confirm that the proposed PIDR architecture outperforms state-of-the-art methods for this task, providing reliable and accurate humidity predictions with a shallow, efficient, and computationally frugal design that aligns with our epistemological constraints. While the NLS-based approach establishes an upper bound for parameter estimation due to its greater precision, the NPE-based method achieves comparable performance, making it a strong alternative for computationally constrained settings (see Appendix B for a detailed power consumption analysis).

Impact of Physical Features in Humidity Prediction

Another key aspect of our analysis was identifying the physical parameters that most strongly influence humidity prediction. In this experiment (Table 3), we assessed the impact of the estimated diffusion coefficient D, the anomalous exponent γ, and temperature T on predictive performance, measuring test RMSE across different feature combinations. As in previous analyses, discharge curves were observed over a 4-second interval.

Both methods achieved their lowest test RMSE when the ANN block was trained on all three features. Specifically, PIDR with NLS yielded a test error of 2.236, while PIDR with NPE achieved 2.463. These results highlight the advantage of leveraging all avail-

Table 4 Ablation study on the impact of temporal length of the input signal $\tilde{u}$ on model performance and the relative average evaluation time required by different parameter estimation methods to extract the model parameters D and γ for each time series. The reported CPU time is averaged over 30 samples randomly sampled over the observed dataset. The results indicate that reducing the signal length to 0.5 seconds has minimal impact on prediction accuracy, still yielding results under the 3% threshold. The NPE method is trained using ground truth parameters estimated at 4 seconds with NLS, which is reflected in the results: NPE produces better separated estimates as it attempts to approximate NLS at 4 seconds despite operating on shorter signals. Additionally, NPE significantly reduces computational time, as it only requires a forward pass through a shallow neural network, lowering evaluation time from seconds to milliseconds. In contrast, NLS is computationally more expensive due to its iterative procedure, which involves solving the PDE. It is interesting to notice that the smallest prediction error is obtained with NPE with a temporal duration of the signal of 2.5 seconds.

		Parameter Estimation Method			
		NLS		NPE	
		Test RMSE	Evaluation Time (CPU)	Test RMSE	Evaluation Time (CPU)
Duration of the Input Signal	0.25 s	4.956 ±1.104	2.468 ±0.819 s	3.851 ±0.847	6.265 ±2.122 $\times 10^{-4}$ s
	0.50 s	2.875 ±0.795	2.635 ±0.695 s	2.757 ±0.563	8.921 ±3.612 $\times 10^{-4}$ s
	1.00s	2.775 ±0.835	2.023 ±0.713 s	2.294 ±0.483	1.028 ±0.625 $\times 10^{-3}$ s
	1.50 s	2.768 ±0.845	3.562 ±0.722 s	2.265 ±0.673	1.331 ±0.405 $\times 10^{-3}$ s
	2.00 s	2.366 ±0.736	4.211 ±0.815 s	3.209 ±0.501	1.759 ±0.832 $\times 10^{-3}$ s
	2.50 s	2.476 ±0.735	4.609 ±0.995 s	**1.914** ±0.632	2.149 ±0.372 $\times 10^{-3}$ s
	3.00 s	3.240 ±0.723	4.899 ±0.474 s	3.178 ±0.894	2.448 ±0.527 $\times 10^{-3}$ s
	3.50 s	2.206 ±0.609	5.484 ±0.695 s	2.243 ±0.652	5.667 ±0.824 $\times 10^{-3}$ s
	4.00 s	2.236 ±0.957	5.955 ±0.968 s	2.463 ±0.744	3.310 ±0.631 $\times 10^{-3}$ s

able physical information for accurate prediction. However, an interesting finding emerges when considering only D and T as input features in the latent space. Given the strong correlation between D and γ (see Figure 4), removing γ still resulted in good predictive performance, with RMSE values of 2.366 for NLS and 2.517 for NPE, both remaining below the 3% threshold. This suggests the potential for further reducing the number of trainable parameters in the overall model, improving efficiency without significantly compromising accuracy.

The primary difference in performance arises at low humidity levels, where γ becomes more informative. In this regime, γ helps differentiate between similar data points, leading to a slight improvement in discrimination when included. Additionally, the results underscore the critical role of temperature in the prediction task. Omitting T significantly degrades performance, reinforcing its importance in conditioning the posterior probability of humidity. Temperature provides essential physical context, influencing diffusion dynamics and ensuring robust predictions across different environmental conditions.

Robustness of the Method To Input Signal Duration A crucial aspect of designing a predictive model for a humidity sensor is evaluating its robustness to reduced observation time. Specifically, we aim to determine the minimum observation duration required before predictive accuracy significantly deteriorates. To assess this, we tested both parameter estimation methods–NLS and NPE–on discharge curves of varying lengths, ranging from 0.25 to 4.0 seconds. Additionally, we measured the average computational time required for parameter estimation per sample on a CPU. While NPE can leverage GPUs for faster inference, we focused on CPU performance, as simpler hardware is often more suitable for real-world deployment.

The results, summarized in Table 4, show that both methods maintain consistently high performance for observation times down to 0.5 seconds, with RMSE values remaining below the acceptable threshold. However, at 0.25 seconds, prediction accuracy declines significantly due to the nearly indistinguishable nature of the early portions of the discharge curves. We also note that the error bars (standard deviations) reveal a significant overlap in the confidence intervals for RMSE across different signal durations. This overlap indicates

that the fluctuations in RMSE with respect to signal duration are due to the inherent stochasticity of the measurements, rather than a deterministic trend.

Beyond accuracy, NPE offers substantial advantages over NLS. In addition to reducing computational cost by up to three orders of magnitude, it allows training on shorter curves while still predicting physical parameters as if the full 4-second observation were available. Specifically, we trained NPE using GMM sampling (Algorithm 1) based on the full 4-second dataset, then truncated the time series to simulate shorter observations. This approach enables NPE to estimate parameters as if it had access to the complete curve, even when provided with limited input.

This property gives NPE a significant advantage: as observation time decreases, NPE consistently outperforms NLS in accuracy while requiring significantly less computation. The best result for NPE is achieved at 2.5 seconds, with a test RMSE of 1.914, whereas NLS reaches its best performance at 3.5 seconds with an RMSE of 2.206. Furthermore, NPE's inference time is in the millisecond range per sample, enabling real-time predictions. This makes it particularly well-suited for deployment in sensor applications and real-world environments where rapid and efficient estimation is critical.

4 Conclusion

In this work, we addressed the challenge of developing an accurate and computationally efficient method for humidity estimation using a novel sensor based on electric discharge dynamics across microelectrodes. We introduced **PIDR** methodology, a novel dimensionality reduction technique that uses the physical model to extract features from higher dimension data. We focused on humidity estimation using the PIDR coupled with a model of anomalous diffusion governing the discharge. Using NPE, effectively trained on synthetically enhanced data guided by a prior GMM derived from initial NLS estimates, we successfully mapped the high-dimensional voltage time series onto a low-dimensional, physically meaningful latent space defined by the anomalous diffusion parameters (D, γ). This reduced representation, combined with temperature readings, served as input to a final, shallow Artificial Neural Network, enabling accurate humidity predictions with low RMSE, even when trained with limited experimental data.

While the presented PIDR approach using NPE effectively leverages the physical model, future work could explore alternative machine learning strategies to potentially enhance the latent space representation or integrate physical constraints more deeply. For instance, Variational Autoencoders (VAE) and generative models could learn latent representations directly from the discharge curves, potentially capturing subtle dynamics not fully described by the D and γ parameters alone. Integrating our prior and posterior knowledge of the parameter distributions could guide the VAE training towards more structured and physically interpretable latent variables. Alternatively, PINNs offer a powerful method to directly embed the governing anomalous diffusion equation within the neural network training loop. This could refine the estimation of physical parameters or even allow for direct humidity prediction while ensuring that the underlying physical laws are respected, possibly eliminating the need for separate synthetic data generation based on external solvers. From an experimental perspective, our study could be extended to further characterize long-term sensor stability. For example, hysteresis is a relevant factor for humidity sensors, especially at the high relative humidity values investigated in our manuscript. In the current study, our data acquisition protocol was unidirectional. Indeed, for each measurement series, the relative humidity was increased from a lower to a higher setpoint. Because the data was collected only along this "adsorption" path, the model was not trained on data from the "desorption" path, i.e., decreasing humidity. Therefore, our current framework does not explicitly account for hysteresis effects, as its training data did not include this phenomenon. However, these effects could become significant during prolonged sensor operation. Moreover, several other factors could be involved, e.g. repeated measurements, especially at high voltages, can cause wear on the electrode materials, leading to a drift in accuracy. Noticeably, both these effects are mitigated in our investigation. Indeed, we allow the system to return to its original state before a successive discharge is performed, and consecutive discharges are limited in number, so that the materials involved are preserved.

As a conclusive remark, the effectiveness of the PIDR methodology highlights the significant benefits of integrating physical knowledge into machine learn-

ing for developing advanced sensing systems. This approach yields models that are not only accurate and interpretable but also inherently frugal. By reducing data dimensionality based on physics and using compact networks, PIDR enables efficient processing crucial for real-world smart sensors and IoT devices, where computational resources, power budgets, and low latency are primary concerns. This computational efficiency also translates directly to improved sustainability, as frugal models require less energy for both training and operation compared to larger architectures, paving the way for greener AI-powered sensing solutions.

Author contributions A.L. proposed the method, developed the Python routines, designed the machine learning framework, and prepared the manuscript and figures. S.B. provided the MATLAB codes for NLS parameter estimation and PDE solving, and contributed to manuscript revision and methodological refinement. P.B. conducted the laboratory measurements and contributed to writing the sections on the laboratory setup and sensors. M.P. and L.R. supervised the project, providing weekly guidance to ensure theoretical rigor and alignment with industrial applications. All authors actively reviewed and contributed to the final version of the manuscript.

Funding Open access funding provided by Politecnico di Torino within the CRUI-CARE Agreement. A.L., S.B. and L.R., as members of GNFM (Gruppo Nazionale per la Fisica Matematica) of INdAM (Istituto Nazionale di Alta Matematica), acknowledge that this work has been performed under the auspices of GNFM of INdAM. A.L. was supported by the Project Piano Nazionale di Ripresa e Resilienza - Next Generation EU (PNRR-NGEU) from Italian Ministry of University and Research (MUR) under Grant DM 117/2023. This work is part of the project NODES which has received funding from the MUR - M4C2 1.5 of PNRR funded by the European Union - NextGenerationEU (Grant agreement no. ECS00000036). Thanks to Mauro Zorzetto, Matteo Rondano and Marco Ferragatta of the Mobility Dep. of Eltek, who designed the electronic circuit of the sensor, for the useful discussions on microgap applications.

Data Availability Statement The data and code used in this study are available upon reasonable request. Access is subject to approval by Eltek S.p.A., as the datasets and sensor specifications are proprietary and not publicly available.

Declarations

Competing interests The authors declare no competing interests.

Appendix A Robustness to Geometric Variations of the Microgap

We here address the potential impact of geometric imperfections on device performance and provide further details on the sensor's physical characteristics to support experimental reproducibility. The following analysis demonstrates the robustness of the proposed PDE-based modeling approach to variations in both the microgap separation and the overall device geometry. A more extensive analysis, including further ablation studies and hysteresis evaluation, is presented in [8].

The sensor architecture, depicted in Figure 8, consists of a chromium (Cr) thin film with a nominal thickness of 100 nm, deposited on an insulating substrate such as silicon dioxide (SiO_2) or alumina (Al_2O_3). Electrical connections are established through aluminum (Al) or gold (Au) contact pads, each 150 nm thick. The inter-electrode separation, or microgap, ranges from 1.5 to 2.5 μm. The lower bound of 1.5 μm represents the minimum distance reliably achievable with our fabrication methodology without incurring a significant risk of short-circuiting, while the stated range accounts for the inevitable fabrication tolerances at this micro-scale.

To evaluate the impact of these geometric variations, we first investigated the effect of different gap separations. Figure 9 shows a comparison of discharge curves obtained from devices with 2 μm and 10 μm gaps. While minor differences in the dynamics are observable, the overall response profile remains highly consistent. Crucially, preliminary investigations show that the proposed PDE model accurately captures the discharge behavior in both cases, demonstrating its robustness to variations in this key parameter.

Furthermore, we assessed the performance of different device geometries that share the same nominal gap distance (2 μm). As shown in Figure 10, tests were conducted using fluids with varying electrical conductivities. Despite small deviations observed with con-

ductive liquids–likely attributable to the degradation of deionized water or the uncontrolled ionic content of tap water–all devices exhibit comparable and predictable behavior, particularly when using insulating liquids. This suggests that minor variations in the electrode shape or layout do not fundamentally alter the device's electrical response.

In summary, these results support the conclusion that, within typical fabrication tolerances, small geometric imperfections do not significantly impact the sensor's discharge dynamics. The proposed PDE model proves to be robust across these variations, a finding that is consistently corroborated by the more extensive studies presented in [8].

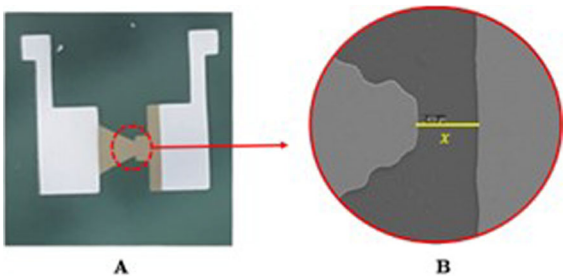

Fig. 8 Geometrical representation of the micro-gap sensor used in our experiments. (*A*) Micro-gap composed of a metal thin film, typically 100 nm of Cr, deposited on an insulating layer (SiO_2 or Al_2O_3), with Al or Au electrical contacts (150 nm thick). (*B*) Zoomed view of the region where the electrode separation is minimal. The spatial domain Ω represents the extent of the micro-gap.

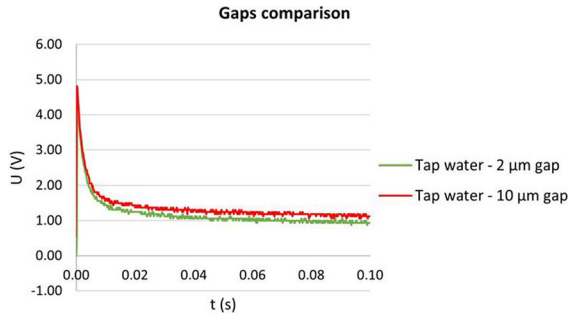

Fig. 9 Tests were conducted by depositing a controlled volume of water onto two micro-gaps with different separations (2 μm and 10 μm). While the resulting curves are very similar, slight differences related to gap size can be observed. Nevertheless, the proposed PDE model provides an accurate fit for both cases.

Appendix B Power Consumption Analysis for Embedded Deployment

In this appendix, we discuss the suitability of the proposed model for resource-constrained environments, by means of a power consumption analysis. Specifically, we estimated the total power required to execute a single forward pass of the NPE model on a representative low-power embedded platform.

B.1 System Configuration

The target system comprises two main components, representative of hardware commonly used in IoT applications:

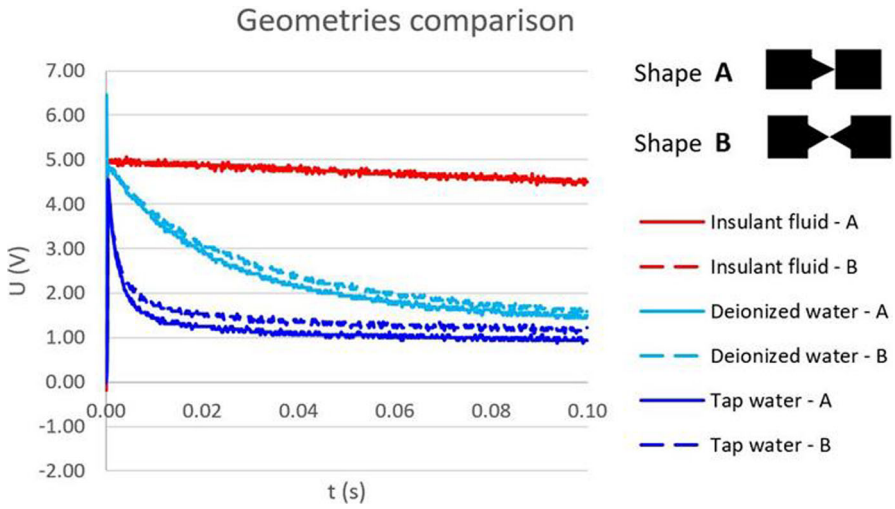

Fig. 10 Tests were carried out on devices with a 2 μm gap using different fluids with varying conductivities. While slight differences are observed for water–possibly due to degradation of deionized water or the uncontrolled conductivity of tap water–the two geometries exhibit consistent behavior, particularly when using insulating liquids with high dielectric strength.

- **Microcontroller (MCU):** STM32F405 series (ARM Cortex-M4), selected for its balance of performance and energy efficiency.
- **External Flash Memory:** Cypress S25FS512S, a 512-Mbit (64-MB) flash device used to store the NPE model weights. The model contains approximately 9.25×10^6 parameters ($\sim$ 37 MB), which exceed the internal memory capacity of most MCUs.

The supply voltage was set to $V_{\text{supply}} = 3.3\,\text{V}$. Current consumption values for each component were taken from the corresponding datasheets.

B.2 Computational Scenarios and Power Estimates

To effectively capture the trade-off between energy efficiency and inference speed, we analyzed two operational scenarios.

Scenario 1: Low-Power, Low-Speed Operation In this setting, the MCU operates at 8 MHz to minimize energy consumption. The estimated current draw is:

$$I_{\text{MCU, low}} \approx 15\,\text{mA}, \tag{B1}$$

$$I_{\text{Flash, active}} \approx 15\,\text{mA}. \tag{B2}$$

Thus, the total current is:

$$I_{\text{total, low}} = I_{\text{MCU, low}} + I_{\text{Flash, active}} = 30\,\text{mA}. \tag{B3}$$

The corresponding power consumption is:

$$P_{\text{low}} = V_{\text{supply}} \times I_{\text{total, low}} = 3.3\,\text{V} \times 30\,\text{mA} = \mathbf{99\,mW}. \tag{B4}$$

Scenario 2: High-Power, High-Speed Operation Here, the MCU operates at its maximum frequency (168 MHz) to achieve fast inference. The estimated currents are:

$$I_{\text{MCU, high}} \approx 105\,\text{mA}, \tag{B5}$$

$$I_{\text{Flash, active}} \approx 15\,\text{mA}, \tag{B6}$$

leading to a total current:

$$I_{\text{total, high}} = I_{\text{MCU, high}} + I_{\text{Flash, active}} = 120\,\text{mA}. \tag{B7}$$

The resulting power consumption is:

$$P_{\text{high}} = V_{\text{supply}} \times I_{\text{total, high}} = 3.3\,\text{V} \times 120\,\text{mA} \approx \mathbf{400\,mW}. \tag{B8}$$

At this operating point, the inference time is reduced to less than one second enabling real-time monitoring and control.

This analysis demonstrates that the proposed model can be deployed on embedded systems under different energy–latency trade-offs. The low-power configuration is suitable when energy efficiency is critical, while the high-power mode supports real-time applications. These results confirm the feasibility of the NPE model for embedded deployment.

References

1. Shuvo, S.P., Hossain, M.J., Roy, B.K., Islam, M.R.: Hybrid noise reduction-based data-driven modeling of relative humidity in khulna, bangladesh. Heliyon **10**(16), 30198 (2024)
2. Uttsha, M.M., Islam, M.R., Roy, B.K., Hossain, M.J.: Enhancing agricultural automation through weather invariant soil parameter prediction using machine learning. Heliyon **10**(7), 28214 (2024)
3. Yang, T., Zhang, Y., Zhang, Y., Li, X., Zhang, Y., Wang, Y., Zhang, D.: Moisture content online detection system based on multi-sensor fusion and convolutional neural network. Front. Plant Sci. **15**, 1289783 (2024)
4. Wawrzynek, E., Baumbauer, C., Arias, A.C.: Characterization and comparison of biodegradable printed capacitive humidity sensors. Sensors **21**(19), 6557 (2021)
5. Vajs, I., Pobar, M., Klančar, G.: Developing relative humidity and temperature corrections for low-cost sensors using machine learning. Sensors **21**(10), 3338 (2021)
6. Pelegrí-Sebastiá, J., García-Diego, F.-J., Fito, P.-J., Valldecabres, L.: Low-cost capacitive humidity sensor for application within flexible rfid labels based on microcontroller systems. IEEE Trans. Instrum. Meas. **61**(2), 545–553 (2012)
7. Pozo, B., Robles, G., Romero, J., Redondo, R.: Supercapacitor electro-mathematical and machine learning modelling for low power applications. Electronics **7**(4), 44 (2018)
8. Bernardi, S., Begnamino, P., Pizzi, M., Rondoni, L.: Anomalous transport models for fluid classification: insights from an experimentally driven approach. Discover Nano **20**(1), 133 (2025)
9. Bishop, C.M.: Mixture density networks. Aston University, Birmingham (1994)
10. Ezhov, I., Lipkova, J., Shit, S., Kofler, F., Collomb, N., Lemasson, B., Barbier, E., Menze, B.: Neural parameters estimation for brain tumor growth modeling. In: Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen,

China, October 13–17, 2019, Proceedings, Part II 22, pp. 787– 795 (2019). Springer
11. Wang, B., Law, M.K., Bermak, A.: A low-cost capacitive relative humidity sensor for food moisture monitoring application. In: 2012 4th Asia Symposium on Quality Electronic Design (ASQED), pp. 111– 114 (2012)
12. Najeeb, M.A., Ahmad, Z., Shakoor, R.A.: Organic thin-film capacitive and resistive humidity sensors: A focus review. Adv. Mater. Interfaces **5**(21), 1800969 (2018)
13. Sankir, N.D.: Flexible electronics: materials and device fabrication. PhD thesis, Virginia Polytechnic Institute and State University (2005)
14. Khan, A.U., Ostermann, J., Ali, M., Khan, M.F., Ahmad, I.: An efficient interface circuit for lossy capacitive sensors. IEEE Trans. Instrum. Meas. **68**(3), 829–836 (2018)
15. Sifuentes, E., Reverter, F., Gasulla, M.: Measuring dynamic signals with direct sensor-to-microcontroller interfaces applied to a magnetoresistive sensor. Sensors **17**(5), 1150 (2017)
16. Reverter, F.: The art of directly interfacing sensors to microcontrollers. Journal of Low Power Electronics and Applications **2**(4), 265–281 (2012)
17. Sifuentes, E., Reverter, F., Gasulla, M.: Seat occupancy detection based on a low-power microcontroller and a single fsr. Sensors **19**(3), 699 (2019)
18. Hidalgo-López, J.A., Reverter, F., Gasulla, M.: Fast calibration methods for resistive sensor readout based on direct interface circuits. Sensors **19**(18), 3871 (2019)
19. Reverter, F., Casas, O.: Direct interface circuit for capacitive humidity sensors. Sens. Actuators, A **143**(2), 315–322 (2008)
20. Medsker, L.R., Jain, L., et al.: Recurrent neural networks. Design and Applications **5**(64–67), 2 (2001)
21. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
22. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. In: NIPS 2014 Workshop on Deep Learning, December 2014 (2014)
23. Eraliev, O., Lee, C.-H.: Performance analysis of time series deep learning models for climate prediction in indoor hydroponic greenhouses at different time intervals. Plants **12**(12), 2316 (2023)
24. Jung, D.-H., Kim, T.-H., Kim, H.-W., Park, M.-H., Lee, J.-H.: A deep learning model to predict evapotranspiration and relative humidity for moisture control in tomato greenhouses. Agronomy **12**(9), 2169 (2022)
25. Kempelis, A., Barzdins, G., Karnitis, G.: Machine learning-based sensor data forecasting for precision evaluation of environmental sensing. In: 2023 IEEE 10th Jubilee Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE), pp. 1– 6 (2023)
26. Vamseekrishna, A., Sridevi, S., Anusha, K., Sandeep, N., Sravani, B.: Prediction of temperature and humidity using iot and machine learning algorithm. In: International Conference on Intelligent and Smart Computing in Data Analytics: ISCDA 2020. Lecture Notes in Networks and Systems, vol. 171, pp. 31– 40 (2021)
27. Aster, R.C., Borchers, B., Thurber, C.H.: Parameter Estimation and Inverse Problems. Elsevier, Amsterdam (2018)
28. Biegler, L., Damiano, J., Blau, G.: Nonlinear parameter estimation: a case study comparison. AIChE J. **32**(1), 29–45 (1986)
29. Yao, L., Sethares, W.A.: Nonlinear parameter estimation via the genetic algorithm. IEEE Trans. Signal Process. **42**(4), 927–935 (1994)
30. Levenberg, K.: A method for the solution of certain nonlinear problems in least squares. Q. Appl. Math. **2**(2), 164–168 (1944)
31. Nolan, S., Smerzi, A., Pezzè, L.: A machine learning approach to bayesian parameter estimation. npj Quantum Information 7(1), 169 (2021)
32. Auld, T., Bridges, M., Hobson, M., Gull, S.: Fast cosmological parameter estimation using neural networks. Monthly Notices of the Royal Astronomical Society: Letters **376**(1), 11–15 (2007)
33. Calderón-Macías, C., Sen, M.K., Stoffa, P.L.: Artificial neural networks for parameter estimation in geophysics [link]. Geophys. Prospect. **48**(1), 21–47 (2000)
34. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press, Oxford (1995)
35. Baumeister, P., Tosi, N.: Exomdn: Rapid characterization of exoplanet interior structures with mixture density networks. Astronomy & Astrophysics **676**, 106 (2023)
36. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Nature Machine Intelligence **1**(12), 457–466 (2019)
37. Jagtap, A.D., Mao, Z., Adams, N., Karniadakis, G.E.: Physics-informed neural networks for inverse problems in supersonic flows. J. Comput. Phys. **466**, 111402 (2022)
38. Zhai, W., Tao, D., Bao, Y.: Parameter estimation and modeling of nonlinear dynamical systems based on runge-kutta physics-informed neural network. Nonlinear Dyn. **111**(22), 21117–21130 (2023)
39. Jiang, X., Wang, D., Chen, X., Zhang, M.: Physics-informed neural network for optical fiber parameter estimation from the nonlinear schrödinger equation. J. Lightwave Technol. **40**(21), 7095–7105 (2022)
40. Bernardi, S., Pizzi, M., Rondoni, L.: Anomalous heat transport and universality in macroscopic diffusion models. J. Therm. Anal. Calorim. **149**(13), 7087–7094 (2024)
41. Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear parameters. J. Soc. Ind. Appl. Math. **11**(2), 431–441 (1963)
42. Moré, J.J.: The levenberg-marquardt algorithm: implementation and theory. In: Numerical Analysis: Proceedings of the Biennial Conference Held at Dundee, June 28–July 1, 1977, pp. 105– 116 (2006). Springer
43. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
44. Geman, S., Bienenstock, E., Doursat, R.: Neural networks and the bias/variance dilemma. Neural Comput. **4**(1), 1–58 (1992)
45. Tomczak, J.M.: Deep generative modeling for neural compression. In: Deep Generative Modeling, pp. 259– 275. Springer, Cham (2024)
46. Villani, C.: Topics in Optimal Transportation vol. 58. American Mathematical Soc.,??? (2021)

47. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
48. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pp. 249– 256 (2010). JMLR Workshop and Conference Proceedings
49. Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983 (2016)
50. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in Neural Information Processing Systems **32** (2019)
51. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
52. Plevris, V., Solorzano, G., Bakas, N.P., Ben Seghier, M.E.A.: Investigation of performance metrics in regression analysis and machine learning-based prediction models (2022)