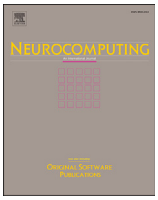
Neurocomputing 670 (2026) 132563

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom



Infinitely deep Bayesian neural network with signature transform

Xiaoyu Yang [a], Peiyi Qiu [b], Yoshinobu Kawahara [a,c,*]

[a] Graduate School of Information Science and Technology, The University of Osaka, Osaka, 565-0871, Japan
[b] The Institute of Infrastructure Inspection, China Academy of Railway Science Co. Ltd., Beijing, 100081, China
[c] RIKEN Center for Advanced Intelligence Project, Tokyo, 103-0027, Japan

HIGHLIGHTS

- We introduce the signature transform to characterise the stochasticity in the current continuous neural networks.
- Our framework integrates the signature transform of stochasticity into the weight of an infinitely Bayesian neural network.
- Our RDE model outperforms current SDE networks in reliability and robustness while capturing the inherent randomness.

ARTICLE INFO

Communicated by X. Zhou

Keywords:
Bayesian neural network
Neural ordinary differential equation
Deep signature

ABSTRACT

This work introduces a novel partially infinitely deep Bayesian neural network, particularly where the weights in each layer are governed by differential equations driven by the signature of Brownian motion. By leveraging the advantages of randomness compression for stochastic processes through signature transforms, our architecture is designed to overcome the limitations of current continuous deep Bayesian neural networks under stochasticity, particularly in terms of reliability and robustness. Additionally, we present a comprehensive mathematical framework that integrates the signature transform of stochasticity into the weight evolution of the Bayesian neural network. To approximate the true posterior, we adopt the approximate Bayesian computation method. Finally, empirical results across image classification tasks (CIFAR-10, CIFAR-10C) demonstrate that our model outperforms existing neural stochastic differential equation models in reliability and robustness while maintaining the memory-efficient training and accuracy of neural stochastic differential equations.

1. Introduction

The residual neural network (ResNet) is a deep learning architecture where the layers learn residual functions with reference to the layer inputs [26], which helps mitigate the vanishing gradient problem to some extent. By extending the number of residual layers in a ResNet to infinity, the output of the network can be represented as the solution to an ordinary differential equation (ODE), resulting in neural ODEs [8]. Neural ODEs offer a more flexible model structure and significantly reduce memory requirements during training by dynamically computing the output. They have been widely applied in areas such as time series analysis, diffusion processes [31,45,54], and causal discovery [3,12]. Building on the foundation of neural ODEs, several continuous neural network frameworks have been proposed. For instance, the augmented neural ODE which is a more expressive model, is empirically more stable and has a lower computational cost [11]. Additionally, a differentiable surrogate for the time cost of standard numerical solvers was proposed by using higher-order derivatives of solution trajectories [32]. Furthermore, improvements have also been proposed in the form of neural stochastic differential equations (Neural SDEs) [38,55], neural operators [34], and neural jump SDEs [30].

To address the challenge caused by the inherent limitations of conventional neural networks, such as overconfidence in predictions, vulnerability to adversarial attacks and lack of interpretability, Bayesian neural networks (BNNs) integrate uncertainty estimation into their predictive capabilities. In contrast to point estimates of neural network parameters, they represent the parameters as probability distributions and estimate them using Bayesian inference. The basis of BNNs is indeed compelling. Firstly, they define flexible hypotheses of predictive functions by providing a prior for all their weights and biases. Then, they perform inference to produce posterior distributions. This probabilistic approach improves the model's generalisability and helps to prevent overfitting (see references [2,6,28,41] for further details).

* Corresponding author at: Graduate School of Information Science and Technology, The University of Osaka, Osaka, 565-0871, Japan.
Email addresses: yangxiaoyu@yahoo.com (X. Yang), peiyiqiu_devin@my.swjtu.edu.cn (P. Qiu), kawahara@ist.osaka-u.ac.jp (Y. Kawahara).

https://doi.org/10.1016/j.neucom.2025.132563
Received 4 June 2025; Received in revised form 3 December 2025; Accepted 27 December 2025
Available online 31 December 2025

By leveraging the advantages of Bayesian inference in the continuous-depth neural network, the continuous-depth Bayesian neural network was introduced, leading to the development of the Bayesian neural ODE [13,53]. This model was further extended by assuming that the network's weights follow an Ornstein-Uhlenbeck stochastic process, which is updated using stochastic variational inference (SVI). In this approach, both the network parameters and the output layer's state are jointly solved using a stochastic differential equation (SDE) solver, making it also known as the BNN-SDE model [52]. This framework supports advanced applications such as the counterfactual ODE model [15], the Bayesian neural controlled differential equation (BNCDE) for estimating treatment effects in continuous time [25], and large eddy simulation (LES) in turbulence closure modeling [4].

However, full stochasticity in Bayesian neural networks (BNNs) can be computationally expensive during both training and inference, and it may not always be necessary [14,33]. Both theoretical and empirical evidence suggests that partial stochasticity can effectively reduce computational complexity while preserving the benefits of uncertainty quantification associated with full stochasticity [49]. The challenge of selecting non-stochastic components within partially stochastic networks is addressed by the variational Bayesian last layers approach [24]. Building on these insights, the partially BNN-SDE model is proposed, which integrates the flexibility of continuous-depth networks with partial stochasticity, offering a balance between computational efficiency and uncertainty estimation [10].

Nevertheless, the aforementioned neural SDE architecture introduces additional randomness, which may compromise the reliability and robustness of the model. Rough path (RP) theory was first proposed by Terry Lyons [39]. RP theory constitutes a new form of integral theory, which is in fact an extension of the Young integral theory. The objective is to resolve the issue of paths which are of "poor regularity", with a particular focus on non-differentiable paths. Indeed, it is a well-established fact that the sample paths of stochastic processes are non-differentiable. To illustrate this point, consider the example of Brownian motion, a classic stochastic process that is non-differentiable. When we consider random problems from the perspective of RPs, we can see that they could provide a pathwise framework; in other words, they could effectively realize randomness compression for stochastic processes. Besides, the signature transform, which is a key tool from RP, could capture a comprehensive set of statistics from paths. This transformation offers several advantages, such as training speed-ups and reduced memory requirements especially when tackling long time series [35,36,40] and irregular time series [9]. Moreover, the signature method has been shown to outperform graph convolutional neural networks (GCNs) in analyzing slow earthquake sequences [47]. Based on this, the signature kernel is proposed. This signature kernel, which is treated as covariances for Gaussian processes, is also applied in Bayesian learning from sequential data [51]. The signature kernel is a symmetric positive semidefinite function defined for any pair of paths, providing a powerful tool for comparing path data [29,44,48]. Additionally, it has been shown that a recurrent neural network (RNN) can be formulated as a signature kernel method within a suitable reproducing kernel Hilbert space (RKHS) [20]. For further developments and applications of RP theory, more details can be found in [5,19].

Considering that RPs can reduce the instability caused by inherent randomness while achieving the effects of stochastic processes, we propose a novel approach that combines the signature transform with partially Bayesian neural networks. Specifically, we assume that the weights in the input layer are governed by differential equations driven by the signature associated with Brownian motion, while the weights in other layers are governed by ordinary differential equations. These weights are updated using the approximate Bayesian computation (ABC) method [17,18]. Hence, our model accounts for the inherent randomness while preserving robustness which other SDE models lack, by leveraging the signature method. The model class that we refer to is called the BNN-RDE model. With this approach, the state of the output layer and the network' parameters can be computed by an Euler solver together. Our contributions to the field are summarised as follows:

- We introduce the signature transform to characterise the stochasticity in continuous-time neural models, extending the current framework of continuous Bayesian neural networks and their descendants.
- We present a comprehensive mathematical framework that incorporates stochasticity into the weight evolution of infinitely deep Bayesian neural networks via the signature transform.
- We demonstrate that our rough differential equation models outperform stochastic networks in terms of reliability and robustness while still capturing the inherent stochasticity, highlighting that a more efficient application of deep signature methods is possible.

In Section 2, we review the neural ODE and SDE models. Section 3 introduces the rough path and related theory. Section 4 presents the proposed model: a partially infinitely deep BNN with the signature framework. Section 5 reports on the performance of our BNN-RDE model in image classification (CIFAR-10) and its robustness to input corruption (CIFAR-10C). Finally, Section 6 contains some conclusions and details of future work.

2. Background

In this section, we recall the neural ODE model and the SDE model.

2.1. Neural ODE

By taking the infinity of residual layers in the ResNet, Chen proposed a neural ODE model [8] which consists of a continuous-time hidden state given by a neural network, mathematically,

$$\frac{dh_t}{dt} = f_\theta(t, h_t) \tag{2.1}$$

with $h_{t_0} = h_0$. Here, h_t is the hidden state at layer t and f_θ should be defined by a neural network with parameter θ and preserve the dimension of h. Note that f_θ should satisfy the Lipschitz condition. Hence, evolving these dynamics over a finite time interval is equivalent to propagating the input through a residual network with infinite depth. Specifically, the adaptive ordinary differential equation (ODE) solvers enable the balancing of evaluation speed and precision.

2.2. Neural SDE

When a dataset is given, there are often many functions that can fit the data well, where a neural network with different parameters will express each one. Unlike traditional neural networks, the Bayesian framework does not provide point estimates of parameters, but rather portrays the uncertainty of these parameters through probability distributions, precisely, learning as posterior inference. To incorporate stochasticity into a continuous-time framework and make it well-suited to modeling infinitely deep networks in a Bayesian setting, the neural SDE model [52] is proposed in which the prior for parameters satisfies the following SDE,

$$dw_t = f_p\left(t, w_t\right) dt + g_p\left(t, w_t\right) db_t. \tag{2.2}$$

Here, b is standard d-dimensional Brownian motion, f_p is assumed to be a linear function for w, and g_p is assumed to be a constant matrix. Then the approximate posterior is parametrized by the following SDE:

$$dw_t = f_q\left(t, w_t, \theta\right) dt + g_q\left(t, w_t\right) db_t. \tag{2.3}$$

f_q is also Lipschtiz and defined as a neural network with parameter θ. Then, μ_p, μ_q are denoted the prior and posterior measures on the path space. Note that g_q does not need to be updated only the constant matrices. They adopted stochastic variational inference (SVI) and

Kullback–Leibler (KL) divergence to parameterise the weights on path space,

$$D_{\mathrm{KL}}\left(\mu_q \| \mu_p\right) = \mathbb{E}_{q_\theta(w)}\left[\int_0^1 \frac{1}{2}\left\|u_\theta\left(t, w_t\right)\right\|_2^2 dt\right]$$

where

$$u_\theta\left(t, w_t\right) = g_p(t, w_t)^{-1}\left[f_q\left(t, w_t; \theta\right) - f_p\left(t, w_t\right)\right].$$

Here, the expectation is taken under the approximate posterior, denoted q_θ. Here, the KL divergence, which aims to measure the difference between two probability distributions over the same random variable, has been popularly used [1,7].

3. Rough path and rough differential equations

RP theory is a new form of integral theory and an extension of Young's theory. The objective is to address the issue of "poor regularity" paths, especially non-differentiable paths. In this section, we review the RP theory.

3.1. Rough path theory

Before introducing our model, we first provide the definition of RP. Here, we use the Hölder exponent to evaluate the level of roughness of the path. A smaller Hölder exponent indicates a rougher path, or "poorer regularity". We will now define the Hölder norm.

Set $[a, b] \subset [0, T]$ and $\Delta_{[a,b]} := \{(s, t) \in \mathbb{R}^2 | a \le s \le t \le b\}$. We write Δ simply when $[a, b] = [0, T]$. For $\eta \in (0, 1]$, denote $C^{\eta-\mathrm{hld}}([a, b], \mathbb{R}^d)$ as the space of η-Hölder continuous functions $\varphi : [a, b] \to \mathbb{R}^d$, equipped with the semi-norm

$$\|\varphi\|_{\eta-\mathrm{hld},[a,b]} := \sup_{a\le s<t\le b} \frac{|\varphi_t - \varphi_s|}{(t-s)^\eta} < \infty.$$

The Banach norm in $C^{\eta-\mathrm{hld}}([a, b], \mathbb{R}^d)$ is $|\varphi_a|_{\mathbb{R}^d} + \|\varphi\|_{\eta-\mathrm{hld},[a,b]}$.

Then, we give the definition of RP.

Definition 3.1. [[21], Section 2] A continuous map

$$\mathbf{X} = \left(1, \mathbf{X}^1, \mathbf{X}^2\right) : \Delta \to T^2(\mathbb{R}^d) = \mathbb{R} \oplus \mathbb{R}^d \oplus (\mathbb{R}^d)^{\otimes 2},$$

is said to be an $\mathbb{R}^d$-valued RP of roughness 2 if it satisfies the following conditions,

(Condition A): For any $s \le u \le t$,

$$\mathbf{X}_{s,t} = \mathbf{X}_{s,u} \otimes \mathbf{X}_{u,t}$$

where $\otimes$ stands for the tensor product.

(Condition B): $\|\mathbf{X}^1\|_{\alpha-\mathrm{hld}} < \infty$ and $\|\mathbf{X}^2\|_{2\alpha-\mathrm{hld}} < \infty$.

Obviously, the 0-th element 1 is omitted and we denote the RP by $\mathbf{X} = (\mathbf{X}^1, \mathbf{X}^2)$. The set of all $\mathbb{R}^d$−valued RPs with $1/3 < \alpha < 1/2$ is denoted by $\Omega_\alpha(\mathbb{R}^d)$. The **Condition A** is equivalent to

$$\mathbf{X}^2_{s,t} - \mathbf{X}^2_{s,u} - \mathbf{X}^2_{u,t} = \mathbf{X}^1_{s,u} \otimes \mathbf{X}^1_{u,t}$$

for any $s \le u \le t$ which is according to the tensor product. We set $|||\mathbf{X}|||_{\alpha-\mathrm{hld}} := \|\mathbf{X}^1\|_{\alpha-\mathrm{hld}} + \|\mathbf{X}^2\|^{1/2}_{2\alpha-\mathrm{hld}}$ and for different RPs $\mathbf{X} = (\mathbf{X}^1, \mathbf{X}^2) \in \Omega_\alpha(\mathbb{R}^d)$ and $\tilde{\mathbf{X}} = (\tilde{\mathbf{X}}^1, \tilde{\mathbf{X}}^2) \in \Omega_\alpha(\mathbb{R}^d)$, we denote its distance by $\rho_\alpha(\star, \cdot)$ which is defined as following:

$$\rho_\alpha(\mathbf{X}, \tilde{\mathbf{X}}) := \|\mathbf{X}^1 - \tilde{\mathbf{X}}^1\|_{\alpha-\mathrm{hld}} + \|\mathbf{X}^2 - \tilde{\mathbf{X}}^2\|^{1/2}_{2\alpha-\mathrm{hld}}.$$

In fact, the RP is the driving path in our model. According to the RP theory, the differential equations driven by RP are considered a type of controlled RP. We then provide the definition of controlled RP and more details can be found in [21].

Definition 3.2. A triple $(Y, Y^\dagger, R^Y)$ is called a controlled path of reference RP $\mathbf{X} \in \Omega_\alpha\left(\mathbb{R}^d\right)$, if

$$Y_t - Y_s = Y^\dagger_s \cdot \mathbf{X}^1_{s,t} + R^Y_{s,t} \quad (0 \le s \le t \le T)$$

where

(1) $Y \in C^{\alpha-\mathrm{hld}}([0, T], \mathbb{R}^n)$,
(2) $Y^\dagger \in C^{\alpha-\mathrm{hld}}\left([0, T], \mathcal{L}\left(\mathbb{R}^d, \mathbb{R}^n\right)\right)$,
(3) $R^Y \in C^{2\alpha-\mathrm{hld}}(\Delta, \mathbb{R}^n)$.

Here, we denote $\mathcal{L}(\mathbb{R}^d, \mathbb{R}^n)$ as the set of bounded linear maps from $\mathbb{R}^d$ to $\mathbb{R}^n$.

Denote the controlled RP space by $\mathcal{Q}^\alpha_{\mathbf{X}}([a, b], \mathbb{R}^n)$ and the semi-norm of the controlled RP $(Y, Y^\dagger, R^Y) \in \mathcal{Q}^\alpha_{\mathbf{X}}([a, b], \mathbb{R}^n)$ as

$$\|\left(Y, Y^\dagger, R^Y\right)\|_{\mathcal{Q}^\alpha_{\mathbf{X}},[a,b]} = \|Y^\dagger\|_{\alpha-\mathrm{hld},[a,b]} + \|R^Y\|_{2\alpha-\mathrm{hld},[a,b]}.$$

In the following, $(Y, Y^\dagger, R^Y)$ is replaced by $(Y, Y^\dagger)$ for simplicity. For different controlled RPs $(Y, Y^\dagger) \in \mathcal{Q}^\alpha_{\mathbf{X}}([a, b], \mathbb{R}^n)$ and $(\tilde{Y}, \tilde{Y}^\dagger) \in \mathcal{Q}^\alpha_{\tilde{\mathbf{X}}}([a, b], \mathbb{R}^n)$, their distance is denoted by $d_{\mathbf{X},\tilde{\mathbf{X}},2\alpha}\left(Y, Y^\dagger; \tilde{Y}, \tilde{Y}^\dagger\right) \overset{\mathrm{def}}{=} \left\|Y^\dagger - \tilde{Y}^\dagger\right\|_{\alpha-\mathrm{hld}} + \left\|R^Y - R^{\tilde{Y}}\right\|_{2\alpha-\mathrm{hld}}$.

Then we give the statement of the solution to the rough differential equation (RDE) driven by RP $\mathbf{X}$.

Proposition 3.3. *Let RP $\mathbf{X} = \left(\mathbf{X}^1, \mathbf{X}^2\right) \in \Omega_\alpha(\mathbb{R}^d)$ and $0 < \beta < \alpha$. Assume f is Lipschitz continuous, $g \in \mathrm{Mat}(n, d)$ and $g \in C^3_b$ (the set of C^3-bounded functions whose derivatives up to order 3 are also bounded). There exists a unique solution to the RDE,*

$$Y_t = \xi + \int_0^t f\left(Y_s\right) ds + \int_0^t g\left(Y_s\right) d\mathbf{X}_s. \tag{3.1}$$

where

$$\int_0^1 g\left(Y_s\right) d\mathbf{X}_s \overset{def}{=} \lim_{|\mathcal{P}|\to 0} \sum_{[s,t]\in\mathcal{P}} \left(g\left(Y_s\right)\mathbf{X}_{s,t} + g\left(Y_s\right)^\dagger \mathbf{X}^2_{s,t}\right),$$

where $g\left(Y_s\right)^\dagger := \nabla g\left(Y_s\right) g\left(Y_s\right)$, $\mathcal{P}$ is the partition of the time interval $[0, 1]$ and the mesh $|\mathcal{P}|$ denotes the length of the largest element of $\mathcal{P}$. Then, $(Y_t, Y_t^\dagger) \in \mathcal{Q}^\beta_{\mathbf{X}}([0, T], \mathbb{R}^n)$ is a controlled RP, where the Gubinelli derivative is $Y_t^\dagger = g(Y_t)$.

Precisely speaking, for every $i \in \{1, \dots, n\}$,

$$\int_0^T \sum_{j=1}^d g_{i,j}\left(Y_s\right) d\mathbf{X}^j_s \overset{def}{=} \lim_{|\mathcal{P}|\to 0} \sum_{[s,t]\in\mathcal{P}} \Big(\sum_{j=1}^d g_{i,j}\left(Y_s\right)\mathbf{X}^{1,j}_{s,t} + \sum_{j=1}^d \sum_{l=1}^d \sum_{k=1}^n \nabla^k g_{i,j}\left(Y_s\right) g_{k,l}\left(Y_s\right)\mathbf{X}^{2,j,l}_{s,t}\Big). \tag{3.2}$$

Proof. Under above conditions, one can deduce the existence and uniqueness of the solution to the RDE with unbounded drift term, which is an application of [46, Theorem 3.1] or [37, Theorem 3.7]. □

Theorem 3.4. *(Universal approximation theorem) Let $\xi \in \mathbb{R}^n$ and $\mathbf{X} = \left(\mathbf{X}^1, \mathbf{X}^2\right) \in \Omega_\alpha(\mathbb{R}^d)$ with $1/3 < \alpha < 1/2$. Assume $(Y; \sigma(Y)) \in \mathcal{Q}^\beta_{\mathbf{X}}([0, T], \mathbb{R}^n)$ with $1/3 < \beta < \alpha$ be the (unique) solution to the following RDE*

$$dY = f(Y_t)dt + \sigma(Y_t)d\mathbf{X}_t, \quad Y_0 = \xi \in \mathbb{R}^n. \tag{3.3}$$

Here, f is globally bounded and Lipschitz continuous and $\sigma \in C^3_b$. Similarly, let $(\tilde{Y}; \sigma(\tilde{Y})) \in \mathcal{Q}^\beta_{\tilde{\mathbf{X}}}([0, T], \mathbb{R}^n)$ with $\tilde{Y}_0 = \tilde{\xi}$. Assume $|||\mathbf{X}|||_{\alpha-\mathrm{hld}}, |||\tilde{\mathbf{X}}|||_{\alpha-\mathrm{hld}} \le M < \infty$. Then, we have the following (local) Lipschitz estimates:

$$d_{\mathbf{X},\tilde{\mathbf{X}},2\beta}(Y, \sigma(Y); \tilde{Y}, \sigma(\tilde{Y})) \le C_M\left(|\xi - \tilde{\xi}| + \rho_\alpha(\mathbf{X}, \tilde{\mathbf{X}})\right). \tag{3.4}$$

and

$$\|Y - \tilde{Y}\|_{\beta-\mathrm{hld}} \le C_M\left(|\xi - \tilde{\xi}| + \rho_\alpha(\mathbf{X}, \tilde{\mathbf{X}})\right). \tag{3.5}$$

Here, $C_M = C(M, \alpha, \beta, L_f, \|\sigma\|_{C^3_b}) > 0$ is a constant depending on $M, \alpha, \beta, L_f, \|\sigma\|_{C^3_b}$. In fact, α, β are constants smaller than $1/2$. L_f and

$\|\sigma\|_{C_b^3}$ are Lipschitz constant for f and bounded constant for the derivatives up to 3 of the coefficient σ respectively. Therefore, the above can be controlled by a uniform constant. Then if $|||\mathbf{X}|||_{\alpha-\text{hld}}, |||\tilde{\mathbf{X}}|||_{\alpha-\text{hld}} \leq M < \infty$, it is not too difficult to choose a constant that makes C_M bounded.

Proof. The proof is a small extension of [21, Theorem 8.5]. □

Proposition 3.5. *Let $\mathbf{B} := (\mathbf{B}^1, \mathbf{B}^2)$ be the RP lifted by standard Brownian motion b. Assume that f_Y satisfies Lipschitz condition and $g_Y \in C_b^3$. Consider the following RDE driven by RP $\mathbf{B}$,*

$$dY_t = f_Y(Y_t)\,dt + g_Y(Y_t)\,d\mathbf{B}_t \quad Y_0 = y \in \mathbb{R}^d. \tag{3.6}$$

In fact, the solution to (3.6) coincides with that to the SDE driven by Brownian motion.

Proof. It is well-known that the sample paths of standard Brownian motion b are α-Hölder continuous ($\alpha \in (\frac{1}{3}, \frac{1}{2})$) almost surely. So the Brownian motion could be lifted to RP $\mathbf{B} := (\mathbf{B}^1, \mathbf{B}^2)$ [[21], Section 3. Here, $\mathbf{B}^1$ is called the first-level path which is the sample path of Brownian motion, and $\mathbf{B}^2$ is called the second-level path which could be well-defined by the Itô or Stratonovich integral. According to the Proposition 3.3, if the unbounded drift coefficient f_Y satisfies the Lipschitz condition and $g_Y \in C_b^3$, there exists a unique global solution to (3.6). According to the result [21, Proposition 5.1], the solution to (3.6) coincides with that to the SDE driven by Brownian motion. □

The above results show that it is reasonable to consider the system driven by Brownian motion under the RP framework.

3.2. Prior process with rough differential equations driven by signature

For any continuous path $X : [0,T] \mapsto \mathbb{R}^d$ that satisfies the Lipschitz condition, one can construct its N-roughness path, denoted $\text{Sig}^N := (S^1, S^2, \dots, S^N)$, where S^k is computed as follows: for any $1 \leq k \leq N$.

$$S_{s,t}^k = \int_{s<t_1<\cdots<t_k<t} dX_{t_1} \otimes \cdots \otimes dX_{t_k} \tag{3.7}$$

that is

$$S_{s,t}^k = \lim_{m(D)\to 0} \sum_{l} \sum_{i=1}^{k-1} S_{s,t_{l-1}}^i \otimes S_{t_{l-1},t_l}^{k-i} \tag{3.8}$$

where $\otimes$ stands for the tensor product and $m(D)$ is the size of the partition. We also provide an example of how to calculate the signature in Example 6.1 of the supplementary material.

According to the Kolmogorov continuity theorem, the sample paths of the Brownian motion are $1/3 < \alpha < 1/2$-Hölder continuous almost surely. Then, based on the Proposition 3.5, to give the signature from Brownian motion, it is sufficient to choose $N := \lfloor \frac{1}{\alpha} \rfloor = 2$ ($\lfloor a \rfloor$ denotes the integer not greater than a). The signature of the linear interpolation of Brownian motion b (still denoted by b) could be constructed via (3.7). Then, due to the result (3.6), it is natural to consider the stochastic system with Brownian motion under the RP framework.

Hence, we assume that the prior process satisfies the following RDE driven by the signature transform of Brownian motion b,

$$dw_t = f_\theta(w_t)\,dt + g_\theta(w_t)\,d\,\text{Sig}^2(b)_t. \tag{3.9}$$

It is clear that the stochasticity is contained in the signature transform in the prior process. Sig^2. Then, the approximate posterior will be given by

$$dw_t = f_\phi(w_t)\,dt + g_\phi(w_t)\,d\,\text{Sig}^2(b)_t. \tag{3.10}$$

We will use the ABC method to approximate the posterior distribution by running repeated simulations and comparing the results with observed data, avoiding the explicit calculation of the likelihood function.

4. Our model: BNN-RDE model

The conventional discrete-depth ResNet model can be written as follows:

$$h_{t+\epsilon} = h_t + \epsilon f_h(h_t, w_t)$$

where $t = 1, \dots, T$ are the layer index, $h_t \in \mathbb{R}^n$ is a vector of hidden unit activations at layer t, $w_t \in \mathbb{R}^m$ is the weight matrix for each layer t and f_h is an arbitrary neural network. Here, h_0 is the input. Then, following the Neural ODE idea [8], it allows to construct the continuous analog of the ResNet: take $\epsilon = 1/T$ and the number of layers $T \to \infty$, then

$$\begin{cases} dw_t = f_w(w_t)\,dt \\ dh_t = f_h(t, h_t, w_t)\,dt, \end{cases} \tag{4.1}$$

with $w_{t_0} = w_0$ and $h_{t_0} = h_0$. The above process yields a differential equation that describes the evolution of the hidden unit as a function of the depth variable t. w_t denotes the standard residual networks, which are characterised by different layerwise "weights" Here, a hypernetwork f_w is introduced that specifies the change in weights as a function of depth and the current weights. Finally, the evolution of the hidden state and weights can be combined using an ordinary differential equation that incorporates the sum of the dimensions of the hidden state and weights.

Prior process Our approach involves performing Bayesian inference on the weight process w_t, which includes specifying a suitable prior process followed by approximate Bayesian computation. Hence, according to Section 3.2, our prior process is assumed to be as in Eq. (3.9). However, full stochasticity is too heavy and not necessary in Bayesian neural networks [49]. Now we assume that only the input layer of the prior process is random but the other layers are fully deterministic. Denote the prior process by $w := (\hat{w}, \tilde{w}) \in \mathbb{R}^m$ where $\hat{w}$ and $\tilde{w}$ denote the input layer and other layers separately. We assume the prior process w follows differential equations driven by signature transform,

$$\begin{cases} d\hat{w}_t = \hat{f}_p(\hat{w}_t)\,dt + \hat{g}_p(\hat{w}_t)\,d\,\text{Sig}_t^2(b) \\ d\tilde{w}_t = \tilde{f}_p(\tilde{w}_t)\,dt, \end{cases} \tag{4.2}$$

where the integral in the right-hand side of (4.2) is pathwise. Here, $\text{Sig}^2 = (\text{Sig}^{2,1}, \text{Sig}^{2,2})$ where $\text{Sig}^{2,1}$ could be constructed by S^1 in (3.8), respectively, $\text{Sig}^{2,2}$ could be constructed by S^2 in (3.8). In other words, it could be rewritten as below:

$$\begin{cases} d\hat{w}_t = \hat{f}_p(\hat{w}_t)\,dt + \hat{g}_p(\hat{w}_t)\,d\,\text{Sig}^{2,1}(b)_t + \nabla_{\hat{w}}\hat{g}_p(\hat{w}_t)\,\hat{g}_p(\hat{w}_t)\,d\,\text{Sig}^{2,2}(b)_t, \\ d\tilde{w}_t = \tilde{f}_p(\tilde{w}_t)\,dt. \end{cases} \tag{4.3}$$

Here, $\text{Sig}^{2,1}(b)$ and $\text{Sig}^{2,2}(b)$ are called the first-level path and second-level path of signature $\text{Sig}^2(b) \in T^2(\mathbb{R}^d)$ separately. Note that $w_0 \in \mathbb{R}^m$.

Then the prior process RDE will be rewritten in the following sense,

$$dw_t^p = f_p(w_t^p)\,dt + g_p(w_t^p)\,d\,\text{Sig}^{2,1}(b)_t + \nabla_{\hat{w}}\hat{g}_p\hat{g}_p(w_t^p)\,d\,\text{Sig}^{2,2}(b)_t \tag{4.4}$$

where

$$f_p := (\tilde{f}_p, \hat{f}_p)^T, \quad g_p := (\hat{g}_p, 0).$$

By combining (4.4) and h in (4.1), our proposed BNN-RDE model can be rewritten as follows,

$$\begin{cases} dw_t^p = f_p(w_t^p)\,dt + g_p(w_t^p)\,d\,\text{Sig}^{2,1}(b)_t + \nabla_{\hat{w}}\hat{g}_p\hat{g}_p d\,\text{Sig}_t^{2,2}(b) \\ dh_t = f_h(t, h_t, w_t)\,dt. \end{cases} \tag{4.5}$$

Considering the partial stochasticity, the number of parameters in our model is significantly.

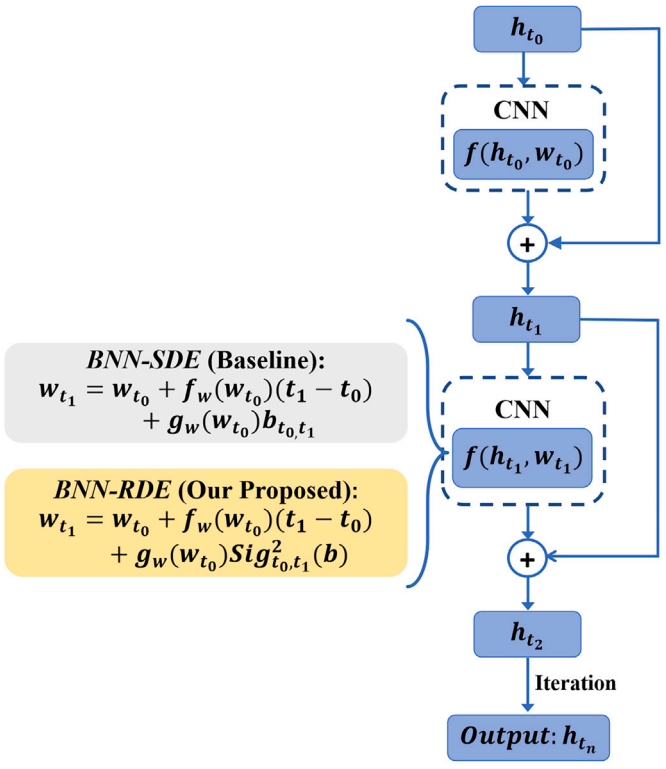


Fig. 1. BNN-SDE, BNN-RDE(our proposed) models.

Then, take $Z_t := (w_t, h_t)^T$ with $Z_0 := (h_0, w_0)^T$, and Z satisfies the following differential equations with signature,

$$dZ_t = F(t, Z_t)dt + G(Z_t)d\,\mathrm{Sig}_t^2(b) \tag{4.6}$$

with

$$F(t, Z) = (f_w(w), f_h(t, h, w))^T, \quad G(Z_t) = (g_w(w), 0)^T,$$

where f_h, f_w, g_w are neural networks. To provide a more intuitive understanding of our proposed model and the baseline BNN-RDE model, a flowchart is provided in Fig. 1.

We can now provide the precise f_w and g_w in the prior process in our BNN-RDE model (4.7) which will be denoted by f_p and g_p.

Proposition 4.1. *Let m_1 and d denote the dimensions of the input layer and b respectively. If we assume that $f_p(w) = -w$ and $\hat{g}_p^{i,j}(w) = \sigma\sin(1 + \xi(\hat{w}))$ with $\xi(w) = \frac{1}{m_1}\sum_{i=1}^{m_1} w^i$ for $i = \{1, \dots, m_1\}$ and $j = \{1, \dots, d\}$. Then for $i \in \{1, \dots, m_1\}$, the prior process (4.3) could be rewritten as below:*

$$\begin{cases} d\hat{w}_t^{p,i} = \hat{f}_p^i\left(\hat{w}_t^p\right)dt + \sum_{j=1}^d \sigma\sin(1 + \xi(\hat{w}))d\,\mathrm{Sig}_t^{2,1,j}(b) \\ \qquad\qquad + \sigma^2\sum_{j=1}^d\sum_{k=1}^d \sin(1 + \xi(\hat{w}_t))\cos(1 + \xi(\hat{w}_t))d\,\mathrm{Sig}_t^{2,2,j,k}(b) \\ d\bar{w}_t = \tilde{f}_p\left(\bar{w}_t\right)dt. \end{cases} \tag{4.7}$$

Proof. By some direct computation, we could deduce that

$$\begin{aligned} d\hat{w}_t^{p,i} &= \hat{f}_p^i\left(\hat{w}_t^p\right)dt + \{\hat{g}_p d\,\mathrm{Sig}_t^2(b)\}^i \\ &= \hat{f}_p^i\left(\hat{w}_t^p\right)dt + \sum_{j=1}^d \hat{g}_p^{i,j}\left(\hat{w}_t^p\right)d\,\mathrm{Sig}_t^{2,1,j}(b) \\ &\quad + \sum_{j=1}^d\sum_{l=1}^d\sum_{k=1}^{m_1} \nabla_{\hat{w}^k}\hat{g}_p^{i,j}\left(\hat{w}_t^p\right)\hat{g}_p^{k,l}\left(\hat{w}_t^p\right)d\,\mathrm{Sig}_t^{2,2,j,l}(b), \end{aligned} \tag{4.8}$$

where $\nabla_{\hat{w}^k}\hat{g}_p^{i,j}$ denotes the k-th partial derivative of function $\hat{g}_p^{i,j}$ with respect to function $\hat{w}$. Since $\hat{f}_p(\hat{w}) = -\hat{w}$, $\hat{g}_p(\hat{w}) = \sigma\sin(1 + \xi(\hat{w}))$, then

Table 1
Algorithm.

Training Algorithm(BNN-RDE)

1: Input: Prior processes w_p, dataset, model $(\hat{w}_t, \tilde{w}_t, h_t)$ and output h, signature of Brownian motion $\mathrm{Sig}^N(b)$,
N : number of simulations, $w_0 = w_p$
2: for $i = 0, \dots, N$ **do**
3: $(\hat{w}_0^0, \tilde{w}_0^0) = (\hat{w}_p, \tilde{w}_p)$
4: $h_0^0 = h_0$
5: **for** $k(t) = 0, \Delta(t), 2\Delta(t), \dots, 1$ **do**
$\hat{w}^i_{k(t)+\Delta(t)} = \hat{w}^i_{k(t)} + f_q(\hat{w}^i_{k(t)})\Delta(t) + g_q(\hat{w}^i_{k(t)})\mathrm{Sig}^2_{k(t),k(t)+\Delta(t)}(b)$
$\bar{w}^i_{k(t)+\Delta(t)} = \bar{w}^i_{k(t)} + f_q(\bar{w}^i_{k(t)})\Delta(t)$
$h^i_{k(t)+\Delta(t)} = h^i_{k(t)} + f^i_h(h^i_{k(t)}, w^i_{k(t)})\Delta(t)$
($\Delta(t)$ denotes the length of the subinterval of the time interval $[0, 1]$)
6: **end for**
7: Compute the loss $L = \rho(h, \hat{h})$ and train the model and update f_q, g_q, f_h.
8: end for
9: Return $(\hat{w}, \bar{w}, h)$

$\nabla_{\hat{w}^k}\hat{g}_p^{i,j} = \frac{\sigma\cos(1+\xi(\hat{w}))}{m_1}$ for all $1 \le k \le m_1$, so Eq. (4.8) could be rewritten as below:

$$\begin{aligned} d\hat{w}_t^{p,i} &= \hat{f}_p^i\left(\hat{w}_t^p\right)dt + \{\hat{g}_p\left(\hat{w}_t^p\right)d\,\mathrm{Sig}_t^2(b)\}^i \\ &= \hat{f}_p^i\left(\hat{w}_t^p\right)dt + \sum_{j=1}^d \sigma\sin(1 + \xi(\hat{w}_t))d\,\mathrm{Sig}_t^{2,1,j}(b) \\ &\quad + \sigma^2\sum_{j=1}^d\sum_{k=1}^d \cos(1 + \xi(\hat{w}_t))\sin(1 + \xi(\hat{w}_t))d\,\mathrm{Sig}_t^{2,2,j,k}(b). \end{aligned} \tag{4.9}$$

The proof is completed. □

Approximate posterior over weights Then the approximate posterior on weights could be defined by another differential equation with different drift terms driven by the signature,

$$\begin{aligned} f_q(w_t) &= NN_\phi - f_p(w_t), \\ g_q(w_t) &= NN_\psi - \hat{g}_p(w_t) \\ G_q(w_t) &= NN_\kappa - \nabla\hat{g}_p(w_t)\hat{g}_p(w_t) \end{aligned} \tag{4.10}$$

where NN_ϕ, NN_ψ and NN_κ are neural networks with parameters ϕ, ψ and κ.

In other words, the approximate posterior should be

$$dw_t^q = f_q(w_t^q)dt + g_q(w_t^q)d\,\mathrm{Sig}^{2,1}(b)_t + G_q(w_t^q)d\,\mathrm{Sig}^{2,2}(b)_t. \tag{4.11}$$

It also could be rewritten as below:

$$\begin{aligned} dw_t^q &= (NN_\phi - f_p(w_t^q))dt + (NN_\psi - \hat{g}_p(w_t^q)\,\mathrm{Sig}^{2,1}(b)_t \\ &\quad + (NN_\kappa - \nabla\hat{g}_p(w_t^q)\hat{g}_p(w_t^q))d\,\mathrm{Sig}^{2,2}(b)_t. \end{aligned} \tag{4.12}$$

Training the network To evaluate our network with a given input x, it must first solve the differential Eq. (4.6). Then, the ABC method is applied to update the approximate posterior of weights. Specifically, given a sampled parameter point $\hat{\theta}$, a simulator dataset $\hat{h}$ is compared with the observed dataset h. If the generated $\hat{h}$ is sufficiently close to the observed data h, the sampled parameter is accepted; otherwise it will be discarded. The distance between the simulator dataset and the observed dataset $\rho(h, \hat{h})$ is defined as a cross-entropy loss function.

Algorithm The algorithm of our BNN-RDE model is shown in Table 1.

5. Experiments

In this section, we report the performance of our partially infinitely deep BNN with the signature framework on image classification and its robustness to input corruption. We also compare it with deterministic

Table 2
Classification accuracy and expected calibration error (ECE) on CIFAR10 (across 3 seeds). Some baseline results are borrowed from [52] and indicated by †. Models are separated into different types: point estimates, stochastic models, and partially stochastic models. There are also discrete-time and continuous-time models.

Model	Accuracy (%) ↑	ECE ($\times 10^{-2}$) ↓
ResNet32†	87.35±0.00	**8.47±0.39**
ODEnet†	**88.30±0.29**	8.71±0.21
HyperODEnet†	87.92±0.46	15.86±1.25
MFVI ResNet32†	86.97±0.00	3.04±0.94
MFVI†	86.48	**1.95**
Deep Ensemble†	89.22	2.79
HMC("gold standard")†	**90.70**	5.94
MFVI ODEnet†	81.59±0.01	3.62±0.40
MFVI HyperODEnet†	80.62±0.00	4.29±1.10
BNN-SDE	87.21±0.32	9.20±1.59
BNN-PSDE	86.93±1.51	**5.17±0.94**
BNN-RDE(Our model)	**87.43±0.27**	**5.48±0.28**

models, as well as infinitely deep BNNs with fully stochastic and partially stochastic structures. We set tanh as an activation function to ensure that f_q and g_q meet the associated conditions. Similar to the BNN-SDE [52] model and BNN-PSDE model [10], we also adopt the initialization method.

5.1. Image classification

Similar to the BNN-SDE [52] and the BNN-PSDE model [10], the hidden state f_h is modified instantaneously by a convolutional neural network (CNN) where the strided convolution and the transposed convolution layer are adopted for downsampling and upsampling respectively. The weights w_t act as filters and biases for these CNN layers. However, different from the above models with SVI, in our model both the drift and diffusion coefficients of the weight w are parameterised in order to minimize the distance between the simulator dataset and the observed dataset where the MLP includes bottleneck layers (e.g. 2–128–2).

We investigated the effectiveness of our proposed BNN-RDE model for training in terms of image classification for CIFAR-10. In particular, a multiscale approach is employed, consisting of several BNN-SDE blocks interspersed with invertible downsampling layers from [16]. The evaluation of the model, including accuracy and expected calibration error [22] is presented in Table 2.

From Table 2, it is found that the proposed BNN-RDE model exhibits better performance in terms of accuracy relative to another stochastic framework, confirming the effectiveness of our proposed approach. Furthermore, the results are consistent with our theoretical analysis below Theorem 3.4 in Section 3. The ECE is a widely used metric to assess whether the output scores of the model can accurately be interpreted as the true probability [42,43]. Reducing the ECE brings the model's output scores closer to the true probabilities, thereby enhancing the model's reliability and applicability [43]. Compared with the BNN-SDE baseline [52], our BNN-RDE model also appears to exhibit obviously superior performance in calibration when the accuracy is relatively close. This is because the signature transformation employed is a kind of pathwise theory, which means that its result almost surely holds, making it capable of compressing the instability inherent in the randomness of stochastic systems. Besides, our BNN-RDE model exhibits a stronger trade-off between performance and calibration than the BNN-PSDE baseline.

The BNN-SDE proposed in [52] has limited practical applicability due to its excessively long training time. The other key point is that our model requires fewer solver steps than the BNN-SDE and BNN-PSDE models. To assess the efficiency of our model, we measure the average time per epoch required for 10,000 CIFAR-10 predictions. Our model requires less training time overall compared to the BNN-SDE model, as shown in Table 3. Using the same Tesla V100-SXM2-32GB server, the overall training time is 26.7% faster than the BNN-SDE model.

Table 3
Epoch time.

Model	SDE	PSDE	RDE
Epoch time	423s	360s	310s

5.2. Robustness to input corruption

By deliberately corrupting the data (e.g., adding noise, blurring, masking, distortion, etc.), corruption experiments assess whether the model can still make accurate predictions despite imperfections or complexities encountered in real-world scenarios. These experiments enable models to learn more robust features that are not solely dependent on perfect data, thus improving their ability to handle uncertainty in practical applications. Common corruption techniques include adding noise (e.g., Gaussian noise, salt-and-pepper noise), blurring, masking, or removing parts of the information, and introducing data distortions (e.g., image rotation, scaling, color adjustments, etc.). We evaluate our BNN-RDE model's robustness using the CIFAR-10C dataset [23] which adds 17 different types of corruptions (e.g. Gaussian noise, pixelation, blur, etc.) with 5 different intensity levels. The result for the average error for each intensity level is shown in Table 4. Even though the BNN-SDE model and our model are trained on such diverse forms or levels of corruption, our BNN-RDE model shows superior robustness to observational noise. Especially as the level of data corruption increases, for instance, when corruption reaches level five—our error rate is two percentage points lower than that of the BNN-SDE model.

5.3. Computational complexity overview

We now provide a breakdown of the time complexities for each model. Let $D = D_h + D_w$ denote the dimension where D_h, D_w denote the dimensions of the hidden state h_t and weights w_t respectively. We also denote the dimension of the input layer of weights w_t by $D_{w-input}$. Then we give the time complexities for BNN-SDE, BNN-PSDE and our model BNN-RDE (Table 5):

1. Time complexity in deterministic part (i.e. induced by dt): Both the BNN-SDE and BNN-PSDE models have the same complexity, which is given by the formula $O(F(D)T)$, where T is the total number of time steps and $F(D)$ represents the cost of evaluating the functions f_w and f_h in (4.5) at each time step. Our model, BNN-RDE, has a complexity of order $O(F(D)T_1)$ for the deterministic part of the Euler discretisation. Here, T_1 is the total number of time steps, which is smaller than T in the BNN-SDE and BNN-PSDE models.

2. Time complexity in stochastic part (i.e. induced by db_t): Both the BNN-SDE and BNN-PSDE models have the same complexity, which is expressed as $\mathcal{O}(D_w T)$ for the stochastic part. Our BNN-RDE model shares the complexity $\mathcal{O}((D_b^2 D_{w-input})T_1)$ for the stochastic part where D_b denotes the dimension of Brownian motion. Here, $(D_b^2 D_{w-input})$ comes from the computation of the signature transform.

The introduction of RPs lies in enhancing the calibration and robustness to input corruption. However, RPs can also lead to an increase in computational dimensions, due to its inherent iterated integral computation, which are also presented at [40]. Therefore, to address this issue while maintaining the calibration and robustness, partial stochasticity is introduced, precisely, only the input layer is assumed as the stochastic process. By combining these two approaches, the proposed BNN-RDE model could enhance the calibration and robustness to input corruption without excessively increasing the computational complexity. In fact, from the Image classification experiment, compared to the standard BNN-SDE model, our BNN-RDE model not only enhances performance and calibration but also reduces computational complexity. In comparison to the BNN-PSDE model, our BNN-RDE model exhibits a stronger trade-off between the performance and calibration. From the input corruption experiment, our proposed BNN-RDE model achieves better robustness. Therefore, although using a smaller $T_1 < T$ helps reduce

Table 4
Error analysis on CIFAR10-C.

Model	Level 1	Level 2	Level 3	Level 4	Level 5
SDE	0.184±0.005	0.238±0.010	0.290±0.014	0.357±0.017	0.449±0.018
PSDE	0.183±0.009	0.232±0.012	0.280±0.009	0.344±0.006	0.430±0.009
RDE(Our model)	**0.176±0.005**	**0.223±0.004**	**0.272±0.006**	**0.339±0.009**	**0.428±0.008**

Table 5
Time Complexity.

Model	Time Complexity
BNN-SDE	$\mathcal{O}((F(D)+D_w)T)$
BNN-PSDE	$\mathcal{O}((F(D)+D_w)T)$
BNN-RDE	$\mathcal{O}((F(D)+(D_b^2 D_{w-input}))T_1)$

the computational complexity in the experimental part, the key reason for the reduction in computational complexity is that $D_b^2 D_{w-input}$ in our BNN-RDE model is much smaller than D_w in baselines.

6. Conclusion and future work

We introduced a novel approach combining the signature transform with Bayesian neural networks to characterise stochasticity in continuous-time neural models. Our model modifies neural ODEs, which represent the continuous form of the ResNet. The most widely recognized application of residual networks is in image classification. Therefore, the comparisons between baselines here are conducted on CIFAR-10, which is a typical dataset for evaluating model performance in image classification. From the experiment on the CIFAR-10 dataset, the BNN-RDE model exhibits better accuracy than another stochastic framework, and a stronger trade-off between performance and calibration than baselines. We also evaluated the robustness of our BNN-RDE model using the CIFAR-10C dataset. The results show that, despite being trained on diverse forms and levels of corruption, our BNN-RDE model is more robust to observational noise than the BNN-SDE model. This experiment confirms the effectiveness of our proposed approach, which is consistent with our theoretical analysis in Section 3.

Due to its inherent properties, the rough path (so-called signature transform) can capture richer information from data and compress the intrinsic randomness within stochastic systems. Its first characteristic has led to current applications primarily in data augmentation for time-series data [40] and learning fractional white noise in neural stochastic differential equations [50]. Its second characteristic enables its use in diffusion models for time series generation [27]. The exploration of rough path applications in machine learning remains in its preliminary stages. This constitutes our first endeavour to implement rough paths in Bayesian neural networks within this paradigm, and the experimental findings suggest encouraging outcomes. Through theoretical analysis, the application of rough paths to stochastic problems or diffusion models is indeed feasible. Subsequent research will concentrate on ascertaining whether this rough path theory can be extended to a greater number of scenarios.

CRediT authorship contribution statement

Xiaoyu Yang: Writing – original draft, Methodology, Investigation. **Peiyi Qiu:** Writing – review & editing, Validation, Software. **Yoshinobu Kawahara:** Writing – review & editing, Supervision, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is partially supported by JSPS KAKENHI Grant Number JP22H00516, JP22H05106, and JST CREST Grant Number PMJCR1913. P.Y. Qiu was partially supported by the Key Project of China Academy of Railway Sciences under Grant Number 2024YJ383.

Appendix A. Supplementary data

Supplementary data for this article can be found online at doi:10.1016/j.neucom.2025.132563.

Data availability

No data was used for the research described in the article.

References

[1] C. Archambeau, M. Opper, Y. Shen, et al., Variational inference for diffusion processes, Adv. Neural Inf. Process. Syst. (2007) 20.
[2] M. Abdar, F. Pourpanah, S. Hussain, et al., A review of uncertainty quantification in deep learning: techniques, applications and challenges, Inf. Fusion 76 (2021) 243–297.
[3] A. Bellot, K. Branson, M. van der Schaar, Consistency of mechanistic causal discovery in continuous-time using neural odes, arXiv preprint arXiv:2105.02522, 2021.
[4] A. Boral, Z.Y. Wan, L. Zepeda-Núñez, et al., Neural ideal large eddy simulation: modeling turbulence with neural stochastic differential equations, Adv. Neural Inf. Process. Syst. 36 (2024).
[5] B. Barancikova, Z. Huang, C. Salvi, SigDiffusions: Score-Based Diffusion Models for Time Series via Log-Signature Embeddings, arXiv preprint arXiv:2406.10354, 2024.
[6] C. Blundell, J. Cornebise, K. Kavukcuoglu, et al., Weight uncertainty in neural network, in: International Conference on Machine Learning, PMLR, 2015, pp. 1613–1622.
[7] P. Cattiaux, C. Léonard, Minimization of the Kullback information of diffusion processes, Ann. IHP Probab. Stat. 30 (1) (1994) 83–132.
[8] R.T.Q. Chen, Y. Rubanova, J. Bettencourt, et al., Neural ordinary differential equations, Adv. Neural Inf. Process. Syst. (2018) 31.
[9] Y. Chen, K. Ren, Y. Wang, et al., Contiformer: continuous-time transformer for irregular time series modeling, Adv. Neural Inf. Process. Syst. 36 (2024).
[10] S. Calvo-Ordonez, M. Meunier, F. Piatti, et al., Partially stochastic infinitely deep Bayesian neural networks, in: International Conference on Machine Learning, PMLR, 2024, pp. 5436–5452.
[11] E. Dupont, A. Doucet, Y.W. Teh, Augmented neural ODES, Adv. Neural Inf. Process. Syst. (2019) 32.
[12] E. De Brouwer, A. Arany, J. Simm, et al., Latent convergent cross mapping, in: International Conference on Learning Representations, 2020.
[13] R. Dandekar, K. Chung, V. Dixit, et al., Bayesian neural ordinary differential equations, arXiv preprint arXiv:2012.07244, 2020.
[14] E. Daxberger, E. Nalisnick, J.U. Allingham, et al., Bayesian deep learning via subnetwork inference, in: International Conference on Machine Learning, PMLR, 2021, pp. 2510–2521.
[15] E. De Brouwer, J. Gonzalez, S. Hyland, Predicting the impact of treatments over time with uncertainty aware neural differential equations, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2022, pp. 4705–4722.
[16] L. Dinh, J. Sohl-Dickstein, S. Bengio, Density estimation using real NVP, in: International Conference on Learning Representations, 2022.
[17] Dyer J., Cannon P., Schmon S.M. Approximate bayesian computation with path signatures. arXiv preprint arXiv:2106.12555, 2021.
[18] P. Del Moral, A. Doucet, A. Jasra, An adaptive sequential monte carlo method for approximate Bayesian computation, Stat. Comput. 22 (2012) 1009–1020.
[19] A. Fermanian, T. Lyons, J. Morrill, et al., New directions in the applications of rough path theory, IEEE BITS Inf. Theory Mag. (2023).
[20] A. Fermanian, P. Marion, J.P. Vert, et al., Framing RNN as a kernel method: a neural ODE approach, Adv. Neural Inf. Process. Syst. 34 (2021) 3121–3134.
[21] P. Friz, M. Hairer, A Course on Rough Paths, Springer International Publishing, 2020.
[22] C. Guo, G. Pleiss, Y. Sun, et al., On calibration of modern neural networks, in: International Conference on Machine Learning, PMLR, 2017, pp. 1321–1330.
[23] D. Hendrycks, T. Dietterich, Benchmarking neural network robustness to common corruptions and perturbations, arXiv preprint arXiv:1903.12261, 2019.

[24] J. Harrison, J. Willes, J. Snoek, Variational bayesian last layers, arXiv preprint arXiv:2404.11599, 2024.
[25] K. Hess, V. Melnychuk, D. Frauen, et al., Bayesian neural controlled differential equations for treatment effect estimation, arXiv preprint arXiv:2310.17463, 2023.
[26] K. He, X. Zhang, S. Ren, et al., Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[27] L. Hodgkinson, C. van der Heide, F. Roosta, et al., Stochastic continuous normalizing flows: training sdes as ODES, in:Uncertainty in Artificial Intelligence, PMLR, 2021, pp. 1130–1140.
[28] P. Izmailov, S. Vikram, M.D. Hoffman, et al., What are Bayesian neural network posteriors really like? , in: International Conference on machine Learning, PMLR, 2021, pp. 4629–4640.
[29] Z. Issa, B. Horvath, M. Lemercier, et al., Non-adversarial training of neural sdes with signature kernel scores, Adv. Neural Inf. Process. Syst. 36 (2024).
[30] J. Jia, A.R. Benson, Neural jump stochastic differential equations, Adv. Neural Inf. Process. Syst. (2019) 32.
[31] M. Jin, H.Y. Koh, Q. Wen, et al., A survey on graph neural networks for time series: forecasting, classification, imputation, and anomaly detection, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, 2024.
[32] J. Kelly, J. Bettencourt, M.J. Johnson, et al., Learning differential equations that are easy to solve, Adv. Neural Inf. Process. Syst. 33 (2020) 4370–4380.
[33] A. Kristiadi, M. Hein, P. Hennig, Being bayesian, even just a bit, fixes overconfidence in RELU networks, in: International Conference on Machine Learning, PMLR, 2020, pp. 5436–5446.
[34] N. Kovachki, Z. Li, B. Liu, et al., Neural operator: learning maps between function spaces with applications to PDES, J. Mach. Learn. Res. 24 (89) (2023) 1–97.
[35] P. Kidger, J. Morrill, J. Foster, et al., Neural controlled differential equations for irregular time series, Adv. Neural Inf. Process. Syst. 33 (2020) 6696–6707.
[36] P. Kidger, P. Bonnier, I. Perez Arribas, et al., Deep signature transforms, Adv. Neural Inf. Process. Syst. (2019) 32.
[37] H.D. Luu, Controlled differential equations as rough integrals, Pure Appl. Funct. Anal. 7 (4) (2022) 1245–1271.
[38] X. Liu, T. Xiao, S. Si, et al., Neural sde: Stabilizing neural ode networks with stochastic noise, arXiv preprint arXiv:1906.02355, 2019.
[39] T. Lyons, Z. Qian, System Control and Rough Paths, Oxford University Press, 2002.
[40] J. Morrill, C. Salvi, P. Kidger, et al., Neural rough differential equations for long time series, in: International Conference on Machine Learning, PMLR, 2021, pp. 7829–7838.
[41] R.M. Neal, Bayesian Learning for Neural Networks, Springer Science & Business Media, 2012.
[42] M.P. Naeini, G. Cooper, M. Hauskrecht, Obtaining well calibrated probabilities using Bayesian binning, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 29, 2015 (1).
[43] N. Posocco, A. Bonnefoy, Estimating expected calibration errors, in: Artificial Neural Networks and Machine Learning–ICANN 2021: 30th International Conference on Artificial Neural Networks, (14-17 September 2021), Proceedings, Part IV 30. Bratislava, Slovakia, Springer International Publishing, 2021, pp. 139–150.
[44] A. Pannier, C. Salvi, A path-dependent PDE solver based on signature kernels, arXiv preprint arXiv:2403.11738, 2024.
[45] Y. Rubanova, R.T.Q. Chen, D.K. Duvenaud, Latent ordinary differential equations for irregularly-sampled time series, Adv. Neural Inf. Process. Syst. (2019) 32.
[46] S. Riedel, M. Scheutzow, Rough differential equations with unbounded drift term, J. Differ. Equ. 262 (1) (2017) 283–312.
[47] H. Riess, M. Veveakis, M.M. Zavlanos, Path Signatures and Graph Neural Networks for Slow Earthquake Analysis: Better Together? arXiv:2402.03558, 2024.
[48] C. Salvi, T. Cass, J. Foster, et al., The signature kernel is the solution of a goursat PDE, SIAM J. Math. Data Sci. 3 (3) (2021) 873–899.
[49] M. Sharma, S. Farquhar, E. Nalisnick, et al., Do Bayesian neural networks need to be fully stochastic? in: International Conference on Artificial Intelligence and Statistics, PMLR, 2023, pp. 7694–7722.
[50] A. Tong, T. Nguyen-Tang, T. Tran, et al., Learning fractional white noises in neural stochastic differential equations, Adv. Neural Inf. Process. Syst. 35 (2022) 37660–37675.
[51] C. Toth, H. Oberhauser, Bayesian learning from sequential data using Gaussian processes with signature covariances, in: International Conference on Machine Learning, PMLR, 2020, pp. 9548–9560.
[52] W. Xu, R.T.Q. Chen, X. Li, et al., Infinitely deep Bayesian neural networks with stochastic differential equations, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2022, pp. 721–738.
[53] C. Yildiz, M. Heinonen, H. Lahdesmaki, Ode2vae: deep generative second order ODES with Bayesian neural networks, Adv. Neural Inf. Process. Syst. (2019) 32.
[54] L. Yang, Z. Zhang, Y. Song, et al., Diffusion models: a comprehensive survey of methods and applications, ACM Comput. Surv. 56 (4) (2023) 1–39.
[55] Y. Zhu, Y.H. Tang, C. Kim, Learning stochastic dynamics with statistics-informed neural network, J. Comput. Phys. 474 (2023) 111819.

Author biography

Xiaoyu Yang received the M.Sc. degree in applied mathematics in 2019 and Ph.D. degree in mathematics from Northwestern Polytechnical University, China in 2023. From 2023 to 2025, She was a specially-appointed Postdoctoral researcher at Kawahara Laboratory, Graduate School of Information Science and Technology, Osaka University, Osaka, Japan. Now she is a JSPS Postdoctoral researcher at Kyushu University, Fukuoka, Japan. Her research interests are Bayesian neural networks, stochastic processes and rough path theory.

Peiyi Qiu received the B.S. degree in internet of things engineering in 2016, and Ph.D. degree in computer and electronic engineering from Southwest Jiaotong University, Chengdu, China, in 2022. From 2020 to 2022, he was a Junior Special Visiting Fellow at Kanaya Laboratory, Kyushu University, Fukuoka, Japan. He is currently a Research Assistant with the Institute of Infrastructure Inspection, China Academy of Railway Science Co. Ltd., Beijing, China. His research interests include scientific machine learning, computational electromagnetics and numerical methods.

Yoshinobu Kawahara received his B.S., M.S. and Ph.D. degrees in Engineering from The University of Tokyo, Japan, in 2003, 2005 and 2008, respectively. He is currently a full professor at Graduate School of Information Science and Technology, The University of Osaka (Osaka, Japan). Additionally, he serves as the team director of Dynamical Systems Learning Team at RIKEN Center for Advanced Intelligence Project (RIKEN AIP). His research interests include statistical machine learning and its applications to scientific and engineering fields. In particular, he is interested in the intermediate field between machine learning and dynamical systems.