Continuum Dropout for Neural Differential Equations

Jonghun Lee[1*], **YongKyung Oh**[2*], **Sungil Kim**[1,3†], **Dong-Young Lim**[1,3†]

[1]Artificial Intelligence Graduate School, Ulsan National Institute of Science and Technology (UNIST), Republic of Korea
[2]Medical & Imaging Informatics (MII) Group, University of California, Los Angeles (UCLA), CA, USA
[3]Department of Industrial Engineering, Ulsan National Institute of Science and Technology (UNIST), Republic of Korea
jh.lee@unist.ac.kr, yongkyungoh@mednet.ucla.edu, {sungil.kim, dlim}@unist.ac.kr

Abstract

Neural Differential Equations (NDEs) excel at modeling continuous-time dynamics, effectively handling challenges such as irregular observations, missing values, and noise. Despite their advantages, NDEs face a fundamental challenge in adopting dropout, a cornerstone of deep learning regularization, making them susceptible to overfitting. To address this research gap, we introduce Continuum Dropout, a universally applicable regularization technique for NDEs built upon the theory of alternating renewal processes. Continuum Dropout formulates the on-off mechanism of dropout as a stochastic process that alternates between active (evolution) and inactive (paused) states in continuous time. This provides a principled approach to prevent overfitting and enhance the generalization capabilities of NDEs. Moreover, Continuum Dropout offers a structured framework to quantify predictive uncertainty via Monte Carlo sampling at test time. Through extensive experiments, we demonstrate that Continuum Dropout outperforms existing regularization methods for NDEs, achieving superior performance on various time series and image classification tasks. It also yields better-calibrated and more trustworthy probability estimates, highlighting its effectiveness for uncertainty-aware modeling.

Code —
https://github.com/jonghun-lee0/Continuum-Dropout
Extended version — https://arxiv.org/abs/2511.10446

1 Introduction

Neural Differential Equations (NDEs) have emerged as a powerful framework for modeling continuous-time dynamics by integrating differential equations with neural networks (Chen et al. 2018; Rubanova, Chen, and Duvenaud 2019; Oh et al. 2025a). This framework has demonstrated superior performance in handling irregular sampling, missing data, and noisy observations across diverse domains such as physics (Greydanus, Dzamba, and Yosinski 2019), finance (Yang et al. 2023), and others (Rubanova, Chen, and Duvenaud 2019; Kidger et al. 2020; Oh, Lim, and Kim 2024, 2025; Oh et al. 2025b).

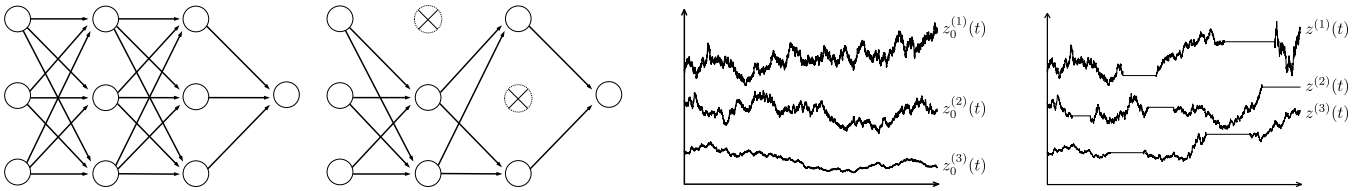
Despite these advantages, NDEs are particularly prone to overfitting, especially when trained on limited data or equipped with complex architectures (Oh, Lim, and Kim 2024). To address this issue, various regularization techniques for NDEs have been explored, including Neural Stochastic Differential Equations (Neural SDEs) (Tzen and Raginsky 2019; Kong, Sun, and Zhang 2020; Oganesyan, Volokhova, and Vetrov 2020), Neural Jump Diffusion model (Liu et al. 2020), STEER (Ghosh et al. 2020), Temporal Adaptive Batch Normalization (TA-BN) (Zheng et al. 2024), and kinetic energy regularization for Neural ODE-based generative models (Finlay et al. 2020). However, these methods are often tailored to specific architectures or fail to offer principled uncertainty quantification. Critically, NDEs face a fundamental challenge in adopting dropout (Srivastava et al. 2014), the cornerstone of regularization in discrete neural networks. For instance, naïvely applying dropout to the drift function results in an ad-hoc solution that fails to capture the underlying continuous-time structure. While the jump diffusion process has been explored for modeling a continuous-time analogue of dropout (Liu et al. 2020), the approach is neither a theoretically faithful analogue nor architecturally general, being applicable only to specific types of NDEs. A summary of the limitations of these existing regularization techniques is provided in Table 1.

	Naïve Dropout	Jump Diffusion	STEER	TA-BN	Continuum Dropout
Tailored for NDEs	×	✓	✓	✓	✓
Discrete Analogue	–	△	×	×	✓
Quantifying Uncertainty	✓	△	×	×	✓
Architectural Generality	✓	×	✓	✓	✓

Table 1: Comprehensive methodological comparison of proposed method with key regularization methods for NDEs

To address this research gap, we introduce Continuum Dropout, a universally applicable regularization technique for NDEs built upon the theory of alternating renewal processes. The key idea is to formulate the on-off mechanism of dropout as a stochastic process that alternates between

*Equal contribution.
†Corresponding authors.



(a) Neural network (b) Neural network w/ dropout (c) NDE, $\mathbf{z}_0(t)$ (d) NDE w/ Continuum Dropout, $\mathbf{z}(t)$

Figure 1: Illustration of dropout in discrete neural networks and continuous-time latent processes: (a) discrete neural network, (b) neural network with dropout, (c) NDE, (d) NDE with Continuum Dropout.

active (evolution) and inactive (paused) states in continuous time. This naturally leads to a modified NDE where the latent dynamics are randomly turned on and off during training. Moreover, in the spirit of Bayesian dropout (Gal and Ghahramani 2016) for discrete networks, Continuum Dropout offers a structured framework to quantify predictive uncertainty via Monte Carlo sampling at test time, enabling more trustworthy and uncertainty-aware modeling. Through extensive experiments, we demonstrate that our method not only outperforms existing regularization techniques but also produces better-calibrated probability estimates, highlighting its practical effectiveness.

2 Preliminaries

Notations. For a $\mathbb{R}^d$-valued vector $\mathbf{x}$, denote the i-th component of $\mathbf{x}$ by $x^{(i)}$ for $i = 1, \ldots, d$. For two matrices $\mathbf{A}$ and $\mathbf{B}$ of the same size, $\mathbf{A} \circ \mathbf{B}$ represents a matrix obtained by element-wise (Hadamard) product of $\mathbf{A}$ and $\mathbf{B}$.

2.1 Problem Statement

Let $\{\mathbf{z}_0(t)\}_{0 \le t \le T}$ be a d_z-dimensional continuous-time dynamical (stochastic) process. In particular, we consider $\mathbf{z}_0(t)$ as a latent process used for various tasks such as prediction, classification, and regression. Let $\mathbf{x}$ denote the d_x-dimensional input data, and $\zeta : \mathbb{R}^{d_x} \to \mathbb{R}^{d_z}$ is an affine function with parameter θ_ζ.

To represent the underlying process $\mathbf{z}_0(t)$, Chen et al. (2018) proposed a Neural ODE as the solution of the following ordinary differential equation

$$\frac{\mathrm{d}\mathbf{z}_0(\mathrm{t})}{\mathrm{dt}} = \gamma(t, \mathbf{z}_0(t); \theta_\gamma) \quad \text{with } \mathbf{z}_0(0) = \zeta(\mathbf{x}; \theta_\zeta), \tag{1}$$

where $0 \le t \le T$, $\gamma(\cdot; \cdot; \theta_\gamma)$ is a neural network parameterized by θ_γ, which is inspired by the following residual connections in ResNet (He et al. 2016):

$$\mathbf{Z}_0(k+1) = \mathbf{Z}_0(k) + \gamma(\mathbf{Z}_0(k); \theta_k), \tag{2}$$

where $\mathbf{Z}_0(k)$ represents the hidden state of ResNet at the k-th layer. See Sander, Ablin, and Peyré (2022) for a detailed discussion of the relationship between ResNets and Neural ODEs.

On the other hand, when it comes to preventing neural networks from overfitting, the most successful and powerful regularization technique in deep learning models is dropout, which randomly deactivates certain neurons during training (Srivastava et al. 2014). Therefore, to further improve the performance of Neural Differential Equations (NDEs), one naturally pose the following question:

Q. How can we incorporate the mechanism of dropout into the NDE framework?

The answer to this question lies in developing a continuous-time analogue of a ResNet with dropout:

$$\mathbf{Z}(k+1) = \mathbf{Z}(k) + \mathbf{I}_k \circ \gamma(\mathbf{Z}(k); \theta_k), \tag{3}$$

where $\mathbf{Z}(k)$ represents the hidden state of ResNet with dropout at the k-th layer and $\mathbb{P}(I_k^{(i)} = 0) = 1 - \mathbb{P}(I_k^{(i)} = 1) = p$, with $\mathbf{I}_k \in \mathbb{R}^{d_z}$. Compared to Equation 2, Equation 3 introduces the Bernoulli variable $\mathbf{I}_k$, which controls the update of the hidden state at step $k+1$. Specifically, when $I_k^{(i)} = 1$, the hidden state evolves as in the standard ResNet, whereas when $I_k^{(i)} = 0$, its evolution is paused.

Figure 1 provides a visual comparison of a standard discrete network, a discrete network with dropout, and their continuous-time counterparts. Consider a standard neural network without dropout, as shown in Figure 1a. When dropout is applied, some neurons are temporarily removed from the network, as shown in Figure 1b. Now, consider the solution $\mathbf{z}_0(t)$ of an NDE, which represents the continuous counterpart of the discrete network shown in Figure 1c. Our method constructs a new continuous-time process $\mathbf{z}(t)$ that emulates dropout by switching between an active state, where it evolves like $\mathbf{z}_0(t)$, and an inactive state, where its evolution is temporarily paused (see Figure 1d). This construction serves as the continuous analogue of the ResNet with dropout in Equation 3, effectively incorporating the dropout mechanism into the NDE framework.

Our key insight is that the behavior of dropout in neural networks in continuous-time can be interpreted as a stochastic process where the system alternates between periods of active (evolution) and inactive (paused) states over random time intervals. This type of stochastic behavior is naturally described by alternating renewal processes. Based on this insight, we provide a more robust and theoretically sound regularization technique for NDEs. In addition, we address important follow-up questions: i) what is the proper definition of the dropout rate in continuous-time settings? and ii) how can the dropout rate be controlled throughout training?

2.2 Alternating Renewal Process

We briefly review the alternating renewal process, which is a key concept in our methodology. This class of regenerative stochastic processes is commonly used to model systems that alternate between two states over time. It has found wide applications in fields such as operations research, queueing

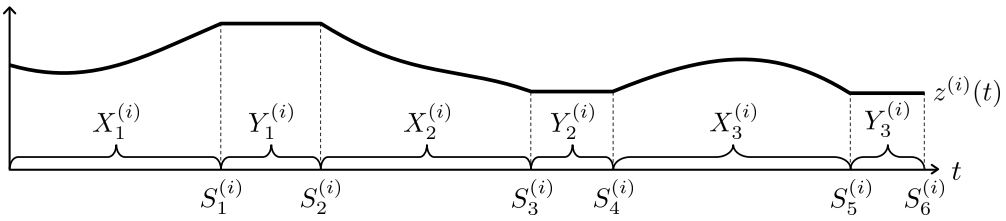


Figure 2: Illustration of i-th component of the latent process $\mathbf{z}(t)$ with Continuum Dropout

theory, and reliability engineering (Stanford 1979; Heath, Resnick, and Samorodnitsky 1998; Pham-Gia and Turkkan 1999; Birolini 1974). For a more detailed explanation, we refer to Appendix A.

Consider a dynamical system that alternates between two (active and inactive) states. Let $\{X_n\}_{n\geq 1}$ be the sequence of i.i.d. random variables with a distribution G representing the lengths of time that the system is in active state, and let $\{Y_n\}_{n\geq 1}$ be the sequence of i.i.d. random variables with a distribution H representing the lengths of time that the system is in inactive state. Assume that X_n and Y_m are independent for any $n \neq m$. However, X_n and Y_n (for the same index n) are not necessarily independent. Moreover, a renewal is defined as the alternation between active and inactive states where the length of each renewal, $\{X_n + Y_n\}_{n\geq 1}$, has a common distribution F. Then, the renewal process associated with $\{X_n, Y_n\}_{n\geq 1}$ is called an *alternating renewal process*. In particular, we focus on an exponential alternating renewal process where G and H are exponentially distributed. This choice is principled because the memoryless property of the exponential distribution ensures that the state transition probability is independent of the elapsed time, which is consistent with how dropout is applied independently of layer depth in discrete networks.

With $S_0 = 0$, define S_n by for each $n \geq 1$,

$$S_n = \sum_{k=1}^{n} T_k, \tag{4}$$

where $T_{2k-1} := X_k$ and $T_{2k} := Y_k$. For $t \geq 0$, let $N(t)$ denote the number of renewals in the interval $[0, t]$, i.e.,

$$N(t) = \sum_{n=1}^{\infty} \mathbf{1}_{\{S_{2n} \leq t\}}. \tag{5}$$

So, $\{N(t)\}_{t\geq 0}$ is the counting process associated with S_{2n}.

Here, S_{2n} indicates the arrival time of the n-th renewal, while S_{2n-1} denotes the start time of the n-th inactive state, which lasts for a duration of Y_n until it alternates to the next active state. Moreover, at time t, the system is in the active state if $S_{2n-2} \leq t < S_{2n-1}$, and in the inactive state if $S_{2n-1} \leq t < S_{2n}$.

3 The Proposed Dropout Method for NDEs

This section introduces Continuum Dropout. First, we present its mathematical formulation based on alternating renewal processes. Then, we explain its practical application, covering the definition of the dropout rate, hyperparameter tuning, and its use for uncertainty quantification.

3.1 Continuum Dropout

Let $\mathbf{z}_0(t) \in \mathbb{R}^{d_z}$, be the solution of an Neural ODE given in Equation 1. While we use a Neural ODE for illustrative purposes throughout this section, our framework is universally applicable to any variant of NDEs. For each $i = 1, 2, \ldots, d_z$, let $\{X_n^{(i)}\}_{n\geq 1}$ and $\{Y_n^{(i)}\}_{n\geq 1}$be sequences of i.i.d. exponential random variables with rates λ_1 and λ_2, representing the lengths of active and inactive periods for the i-th component of the latent process, respectively. These time periods define an dropout indicator function $\mathbf{I}_{\lambda_1,\lambda_2}(t) \in \mathbb{R}^{d_z}$, whose components switch between 1 (active) and 0 (inactive) according to the alternating renewal process defined in Equation 4 and Equation 5. That is, the i-th component of $\mathbf{I}_{\lambda_1,\lambda_2}(t) \in \mathbb{R}^{d_z}$ is given by

$$I^{(i)}_{\lambda_1,\lambda_2}(t) = \begin{cases} 1 & \text{if } S^{(i)}_{2n} \leq t < S^{(i)}_{2n+1}, \\ 0 & \text{if } S^{(i)}_{2n+1} \leq t < S^{(i)}_{2n+2}, \end{cases} \quad \text{for all } n \geq 0.$$

Then, the dynamics of a latent process $\mathbf{z}(t)$ with Continuum Dropout are governed by the following differential equation:

$$\frac{\mathrm{d}\mathbf{z}(t)}{\mathrm{d}t} = \mathbf{I}_{\lambda_1,\lambda_2}(t) \circ \gamma(t, \mathbf{z}(t); \theta_\gamma), \tag{6}$$

with $\mathbf{z}(0) = \mathbf{z}_0(0)$. The solution to Equation 6 can be understood through its integral form. During an active period, for $S^{(i)}_{2n} \leq t < S^{(i)}_{2n+1}$, the trajectory $z^{(i)}$ shares the same differential equation as the original latent process $z_0^{(i)}$. That is, its solution evolves over this interval as follows:

$$z^{(i)}(t) = z^{(i)}(S^{(i)}_{2n}) + \int_{S^{(i)}_{2n}}^{t} \gamma^{(i)}(s, z(s); \theta_\gamma)\, \mathrm{d}s.$$

In contrast, during an inactive period, for $S^{(i)}_{2n+1} \leq t < S^{(i)}_{2n+2}$, its evolution is paused and the state remains at the fixed value $z^{(i)}(S^{(i)}_{2n+1})$, i.e., $z^{(i)}(t) = z^{(i)}(S^{(i)}_{2n+1})$. Figure 2 provides a visual example of a trajectory $z^{(i)}(t)$.

3.2 Dropout Rate

In discrete neural networks, the dropout technique has a hyperparameter called the dropout rate p. Specifically, if the dropout rate is p, then each neuron has a probability of being deactivated for each training iteration. However, in the continuous setting of NDEs, the traditional concept of the dropout rate cannot be directly applicable because each neuron in NDEs represents the value of a continuous-time process at a specific time rather than a countable entity. Therefore, the

concept of the dropout rate needs to be redefined to fit the continuous version.

To this end, we define the continuous-time dropout rate by drawing a direct analogy to its discrete counterpart. In a discrete network, the final prediction is based on the state of the final layer, where each neuron is deactivated with probability p. In an NDE, the final prediction is similarly calculated based on the latent state at the terminal time T. A natural and consistent definition for the dropout rate p is therefore the probability that a component of the latent process is in the inactive state at this terminal time T. This concept is formally known as *instantaneous availability* in renewal theory, which is both theoretically grounded and analytically tractable. Thus, in Continuum Dropout, the dropout rate p is defined as $p = 1 - A(T)$ where $A(T) := \mathbb{P}(\{z^{(i)}(T) \text{ is in the active state}\})$.

This probability $A(T)$ is closely associated with the intensity parameters (λ_1, λ_2) of the alternating renewal process, which determine the expected lengths of the active and inactive periods, respectively. Consequently, the dropout rate p can be expressed in terms of (λ_1, λ_2) in the following theorem.

Theorem 3.1. *For fixed $T > 0$, let $\{\mathbf{z}(t)\}_{0\le t\le T}$ be the latent process with Continuum Dropout where the active and inactive period lengths, $\{X_n^{(i)}\}_{n\ge1}$ and $\{Y_n^{(i)}\}_{n\ge1}$, are i.i.d. exponential random variables with rates λ_1 and λ_2, respectively. Then, the dropout rate $p \in (0,1)$ is given by*

$$p = \frac{\lambda_1}{\lambda_1+\lambda_2}\left(1 - e^{-(\lambda_1+\lambda_2)T}\right). \tag{7}$$

The proof for Theorem 3.1 can be found in Appendix B. Given the desired level of dropout rate p, there are infinitely many possible pairs (λ_1, λ_2) that satisfy Equation 7. To uniquely determine (λ_1, λ_2), we impose a second condition on the expected number of renewals of $\mathbf{z}(t)$ in the interval $[0, T]$, denoted as $m := \mathbb{E}[N^{(i)}(T)]$[1]. In other words, m represents the average number of repetitions of the active and inactive states in the interval $[0, T]$. Consequently, the user-specified pair (p, m) of Continuum Dropout provides two constraints to uniquely determine the intensity parameters (λ_1, λ_2), as stated in the following Corollary.

Corollary 3.2. *Let p be the dropout rate and m be the expected number of renewals of $\mathbf{z}(t)$ in the interval $[0, T]$. Given $p \in (0,1)$ and $m > 0$, the hyperparameters λ_1 and λ_2 of Continuum Dropout can be determined by solving the following system of nonlinear equations:*

$$\begin{cases} p = \dfrac{\lambda_1}{\lambda_1+\lambda_2}\left(1 - e^{-(\lambda_1+\lambda_2)T}\right), \\ m = \dfrac{\lambda_1\lambda_2}{\lambda_1+\lambda_2}T - \dfrac{\lambda_1\lambda_2}{(\lambda_1+\lambda_2)^2}\left(1 - e^{-(\lambda_1+\lambda_2)T}\right). \end{cases} \tag{8}$$

In particular, for large T, λ_1 and λ_2 can be approximated by

$$\lambda_1 \approx \frac{m}{(1-p)\,T}, \quad \lambda_2 \approx \frac{m}{pT}.$$

[1]Note that $\mathbb{E}[N^{(1)}(T)] = \mathbb{E}[N^{(2)}(T)] = \cdots = \mathbb{E}[N^{(d_z)}(T)]$ since $\{X_n^{(i)}\}_{n\ge1}$ and $\{Y_n^{(i)}\}_{n\ge1}$ have common distributions $\text{Exp}(\lambda_1)$ and $\text{Exp}(\lambda_2)$, respectively, for all i.

Algorithm 1: Training and Testing with Continuum Dropout

Input: training data $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N_{\text{train}}}$, testing data $\{\mathbf{x}_i^\dagger\}_{i=1}^{N_{\text{test}}}$, time interval $[0, T]$, hyperparameters $(p, m) \in [0,1) \times \mathbb{R}^+$, number of epochs N_{epochs}, number of MC simulations N_{MC}
Compute: (λ_1, λ_2) from (p, m) according to Equation 8
Initialize: Network parameters $\theta = [\theta_\zeta, \theta_\gamma, \theta_\sigma, \theta_{\text{MLP}}]$
Training Phase

for epoch $= 1$ **to** N_{epochs} **do**
  for each $(\mathbf{x}, \mathbf{y}) \in \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N_{\text{train}}}$ **do**
    $\mathbf{z}(0) \leftarrow \zeta(\mathbf{x}; \theta_\zeta)$
    $\mathbf{z}(T) \leftarrow \text{ODE\_Solve}\Big(\mathbf{I}_{\lambda_1,\lambda_2} \circ \gamma_\theta, \mathbf{z}(0), [0, T]\Big)$[2]
    $\mathbf{y}_{\text{pred}} \leftarrow \text{MLP}(\mathbf{z}(T); \theta_{\text{MLP}})$
    Compute loss $\mathcal{L}(\mathbf{y}_{\text{pred}}, \mathbf{y})$
  end for
  Compute gradients $\nabla_\theta \mathcal{L}$ and update θ using optimizer
end for

Test Phase

for each $\mathbf{x}^\dagger \in \{\mathbf{x}_i^\dagger\}_{i=1}^{N_{\text{test}}}$ **do**
  $\mathbf{z}(0) \leftarrow \zeta(\mathbf{x}^\dagger; \theta_\zeta)$
  Initialize $\mathbf{z}_{\text{sum}} \leftarrow \mathbf{0}$
  for $j = 1$ **to** N_{MC} **do** ▷ MC Simulation
    $\mathbf{z}_j(T) \leftarrow \text{ODE\_Solve}\Big(\mathbf{I}_{\lambda_1,\lambda_2} \circ \gamma_\theta, \mathbf{z}(0), [0, T]\Big)$
    $\mathbf{z}_{\text{sum}} \leftarrow \mathbf{z}_{\text{sum}} + \mathbf{z}_j(T)$
  end for
  $\bar{\mathbf{z}}(T) \leftarrow \mathbf{z}_{\text{sum}}/N_{\text{MC}}$
  $\mathbf{y}_{\text{pred}} \leftarrow \text{MLP}(\bar{\mathbf{z}}(T); \theta_{\text{MLP}})$
end for
return $\{\mathbf{y}_{\text{pred}}\}_{i=1}^{N_{\text{test}}}$

The proof for Corollary 3.2 can be found in Appendix B. While standard dropout has a single hyperparameter, the dropout rate p, the proposed Continuum Dropout intrdocues two hyperparameters, p and m. Nevertheless, our sensitivity analysis summarized in Appendix F.1 indicates that the performance of Continuum Dropout is not highly sensitive to the choice of m, which reduces the time and effort required for hyperparameter tuning. Moreover, a detailed explanation of the parameters (p, m) for the trajectories with Continuum Dropout is provided in Figure 10 of Appendix F.1.

3.3 Uncertainty Quantification in Continuum Dropout

A key strength of our framework is its natural capacity to quantify predictive uncertainty, particularly epistemic uncertainty, driven by the stochasticity of Continuum Dropout. While many NDEs such as Neural ODE (Chen et al. 2018) and Neural CDE (Kidger et al. 2020) generate a single and deterministic latent trajectory for a given input, applying Continuum Droput transforms the dynamics into a stochastic process. This inherent stochasticity, which can be also retained at test time, allows for uncertainty quantification in the spirit of Monte Carlo Dropout (Gal and Ghahramani 2016).

The procedure involves performing N_{MC} stochastic forward passes for a single input. In each pass, a new realization of the dropout indicator function $\mathbf{I}_{\lambda_1,\lambda_2}(t)$ is sampled from the alternating renewal process. Since this indicator function

[2]For brevity, we denote $\gamma(\cdot, \cdot; \theta_\gamma)$ simply as γ_θ.

directly modifies the NDE's dynamics, each sample yields a distinct latent trajectory $\{\mathbf{z}_j(T)\}_{j=1}^{N_{\text{MC}}}$. This collection forms an empirical distribution over the model's prediction for the given input. Then, we use the predictive mean as the point estimator for the prediction:

$$\bar{\mathbf{z}}(T) \;=\; \frac{1}{N_{\text{MC}}} \sum_{j=1}^{N_{\text{MC}}} \mathbf{z}_j(T),$$

and use the sample covariance as a proxy for the model's uncertainty:

$$\widehat{\text{Var}}\big[\mathbf{z}(T)\big] \;=\; \frac{1}{N_{\text{MC}}-1} \sum_{j=1}^{N_{\text{MC}}} \big(\mathbf{z}_j(T)-\bar{\mathbf{z}}(T)\big)\big(\mathbf{z}_j(T)-\bar{\mathbf{z}}(T)\big)^{\top}.$$

The point estimate $\bar{\mathbf{z}}(T)$ is then passed through the output MLP to obtain the final prediction.

As we demonstrate in Section 4.4, we empirically observe that as few as $N_{\text{MC}} \geq 5$ simulations are sufficient to obtain stable performance. The complete procedure is summarized in Algorithm 1.

3.4 Comparison of Alternating Renewal Processes and Jump Diffusion Processes in Dropout Modeling

While the use of a jump diffusion process to model dropout for NDEs is an interesting direction (Liu et al. 2020), we argue that it does not serve as a faithful continuous-time analogue. As discussed in Appendix G.1, the key issue lies in its discretization: the discrete-time version of their proposed process does not recover the standard dropout formula for ResNets with dropout in Equation 3. This theoretical mismatch suggests a fundamental inconsistency in its modeling of the dropout mechanism. In contrast, the discretization of Continuum Dropout, by construction, aligns with Equation 3, providing a more precise representation of the dropout mechanism.

Beyond theoretical misalignment, alternating renewal processes offer significant practical advantages over the jump diffusion model. First, the jump diffusion approach is highly sensitive to the dropout rate, improving performance only for very small values ($p < 0.1$), which deviates from standard practice and requires extensive tuning (see Appendix G.2). On the other hand, Continuum Dropout provides robust performance improvements across a wide range of typical dropout rates. Second, Continuum Dropout consistently outperforms the jump diffusion model in various settings. Lastly, Continuum Dropout is universally applicable to any variant of NDEs. However, the jump diffusion model is limited to specific NDE structures, as detailed in Appendix G.3.

4 Experiments

We perform numerical experiments using real-world datasets to evaluate the effectiveness of the proposed method (Continuum Dropout) in NDEs. First, we compare the proposed method with several regularization methods in Section 4.1. Then, we assess the performance of the proposed method across various NDEs in Section 4.2. In Section 4.3, we examine Continuum Dropout's calibration results to verify its uncertainty-aware modeling capability. Finally, Section 4.4 analyzes how the number of Monte Carlo samples affects both accuracy and computational cost. For Continuum Dropout, we search for optimal hyperparameters among the dropout rate $p \in [0.1, 0.2, 0.3, 0.4, 0.5]$ and the expected number of renewals $m \in [5, 10, 50, 100]$. Throughout our experiments, we use five Monte Carlo samples at test time. A detailed analysis of the impact of MC sample size during the test phase of Continuum Dropout is presented in Section 4.4. Further details of datasets and experimental settings are summarized in Appendix C and Appendix D, respectively.

4.1 Superior Performance Over Existing Regularization Methods

We compare the proposed method (Continuum Dropout) with existing regularization methods for NDEs, using Neural ODE as the baseline. Specifically, we consider **Naïve Dropout for Drift Network** and **Naïve Dropout for MLP Classifier**, where naïve dropout (Srivastava et al. 2014) is applied to the drift network and the MLP classifier, respectively. Additionally, we also consider **Jump Diffusion model** (Liu et al. 2020), **STEER** (Ghosh et al. 2020), and **Temporal Adaptive Batch Normalization (TA-BN)** (Zheng et al. 2024). We compute the performance of both with and without dropout noise during the test phase for Jump Diffusion model, following their experimental setup. TTN refers to the use of dropout noise not only during training but also during the test phase. Table 2 and 3 show the classification performance of various regularization methods on time series and image datasets, respectively. We follow the image dataset selection used in previous studies on regularization methods. The performance of each method is evaluated in terms of top-5 accuracy for CIFAR-100 and top-1 accuracy for the other datasets. Continuum Dropout consistently demonstrates superior performance over existing regularization methods across various datasets.

4.2 Broad Applicability in NDEs

We investigate whether the proposed method (Continuum Dropout) can demonstrate consistent improvements across various NDEs. To evaluate this, we conduct experiments on both time series and image classification tasks. Notably, the time series classification tasks considered here involve challenges such as irregular observations, label imbalance, and missing values where naïve Neural ODE and Neural SDE tend to struggle.

For time series classification tasks, we consider the following NDEs: Neural ODEs (**GRU-ODE** (De Brouwer et al. 2019), and **ODE-RNN** (Rubanova, Chen, and Duvenaud 2019)), Neural CDEs (**Neural CDE** (Kidger et al. 2020) and **ANCDE** (Jhin et al. 2023)), Neural SDEs (**Neural LSDE**, **Neural LNSDE**, and **Neural GSDE** (Oh, Lim, and Kim 2024)). Note that Neural LSDE, Neural LNSDE, and Neural GSDE are state-of-the-art models for the time series classification tasks under consideration. We apply the proposed method to NDEs and evaluate its performance. Table 4 shows classification accuracy of Neural CDE, ANCDE, Neural LSDE, Neural LSDE, Neural LNSDE, Neural GSDE,

Regularization Methods	SmoothSubspace	ArticularyWordRecognition	ERing	RacketSports
Baseline (Neural ODE)	0.569 (0.040)	0.859 (0.005)	0.839 (0.018)	0.565 (0.065)
Naïve Dropout for Drift Network	0.594 (0.016)	0.862 (0.014)	0.844 (0.031)	0.598 (0.045)
Naïve Dropout for MLP Classifier	0.600 (0.067)	0.860 (0.006)	0.856 (0.078)	0.554 (0.076)
Jump Diffusion (Liu et al. 2020)	0.617 (0.043)	0.871 (0.054)	0.861 (0.064)	0.609 (0.031)
Jump Diffusion + TTN (Liu et al. 2020)	0.606 (0.018)	0.876 (0.025)	0.878 (0.025)	0.598 (0.076)
STEER (Ghosh et al. 2020)	0.578 (0.056)	0.871 (0.036)	0.850 (0.029)	0.592 (0.054)
TA-BN (Zheng et al. 2024)	0.578 (0.035)	0.873 (0.023)	0.859 (0.023)	0.587 (0.035)
Continuum Dropout (ours)	**0.629 (0.022)**	**0.884 (0.012)**	**0.881 (0.023)**	**0.619 (0.033)**

Table 2: Performance of various regularization methods on time series classification

Regularization Methods	CIFAR-100	CIFAR-10	STL-10	SVHN
Baseline (Neural ODE)	0.745 (0.012)	0.739 (0.008)	0.707 (0.007)	0.913 (0.004)
Naïve Dropout for Drift Network	0.759 (0.004)	0.749 (0.017)	0.708 (0.002)	0.917 (0.004)
Naïve Dropout for MLP Classifier	0.750 (0.006)	0.751 (0.007)	0.707 (0.004)	0.923 (0.001)
Jump Diffusion (Liu et al. 2020)	0.761 (0.005)	0.750 (0.004)	0.711 (0.002)	0.916 (0.004)
Jump Diffusion + TTN (Liu et al. 2020)	0.760 (0.003)	0.750 (0.005)	0.709 (0.003)	0.917 (0.005)
STEER (Ghosh et al. 2020)	0.756 (0.006)	0.747 (0.008)	0.710 (0.003)	0.915 (0.003)
TA-BN (Zheng et al. 2024)	0.758 (0.003)	0.762 (0.007)	0.701 (0.007)	0.915 (0.004)
Continuum Dropout (ours)	**0.762 (0.002)**	**0.765 (0.007)**	**0.719 (0.005)**	**0.925 (0.004)**

Table 3: Performance of various regularization methods on image classification

with and without Continuum Dropout, on Speech Commands. GRU-ODE and ODE-RNN are excluded due to their failure in training (Kidger et al. 2020). For PhysioNet Sepsis, Table 5 presents the AUROC of various NDEs, with and without Continuum Dropout, under different Observation Intensity (OI) settings. Moreover, we mark "$*$" and "$**$" to represent the statistical significance of the differences between NDEs with and without Continuum Dropout using two-sample t-tests, where p-values are less than 0.05 and 0.01, respectively. As shown in Table 4 and Table 5, the proposed method improves the performance of all NDEs, and most of these improvements are statistically significant. In particular, we emphasize that the performance of the current state-of-the-art models, Neural LSDE, Neural LNSDE, and Neural GSDE, can be further improved with proposed method.

Models	Continuum Dropout	Test Accuracy
Neural CDE	X	0.910 (0.005)
	O	**0.940 (0.001)****
ANCDE	X	0.760 (0.003)
	O	**0.794 (0.003)****
Neural LSDE	X	0.927 (0.004)
	O	**0.932 (0.000)***
Neural LNSDE	X	0.923 (0.001)
	O	**0.932 (0.001)****
Neural GSDE	X	0.913 (0.001)
	O	**0.927 (0.001)****

Table 4: Accuracy on Speech Commands

For image classification tasks, we consider **Neural ODE** (Chen et al. 2018) and **Neural SDEs** (Tzen and Raginsky 2019; Liu et al. 2020; Kong, Sun, and Zhang 2020) (Neural SDE with additive noise and Neural SDE with multiplicative noise). Other variants of NDEs are not suitable for image classification since the concept of a controlled path is designed for handling time series data.

Models	Continuum Dropout	Test AUROC	
		OI	No OI
GRU-ODE	X	0.852 (0.010)	0.771 (0.024)
	O	**0.875 (0.006)****	**0.808 (0.010)***
ODE-RNN	X	0.874 (0.016)	0.833 (0.020)
	O	**0.893 (0.004)***	**0.851 (0.013)**
Neural CDE	X	0.909 (0.006)	0.841 (0.007)
	O	**0.918 (0.001)***	**0.860 (0.003)****
ANCDE	X	0.900 (0.002)	0.823 (0.003)
	O	**0.910 (0.001)****	**0.840 (0.005)****
Neural LSDE	X	0.909 (0.004)	0.879 (0.008)
	O	**0.927 (0.002)****	**0.892 (0.003)****
Neural LNSDE	X	0.911 (0.002)	0.881 (0.002)
	O	**0.930 (0.001)****	**0.891 (0.003)****
Neural GSDE	X	0.909 (0.001)	0.884 (0.002)
	O	**0.929 (0.003)****	**0.890 (0.002)****

Table 5: AUROC on PhysioNet Sepsis

Table 6 presents the performance of Neural ODE, Neural SDE (additive), and Neural SDE (multiplicative) with and without Continuum Dropout on the four image datasets. The experimental results suggest that proposed method consistently enhances the performance of all NDEs across all datasets with statistical significance.

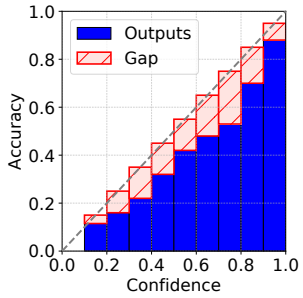
4.3 Capability for Uncertainty-Aware Modeling

While the primary motivation for Continuum Dropout is regularization, its design provides a computationally efficient framework for uncertainty quantification (UQ). As explained in Section 3.3, we can obtain an empirical predictive distribution via Monte Carlo sampling by retaining the stochastic mechanism during inference. Our goal here is not to claim state-of-the-art performance against specialized Bayesian methods in UQ tasks, but rather to demonstrate that our approach yields a crucial UQ benefit, improved calibration, as a natural byproduct.

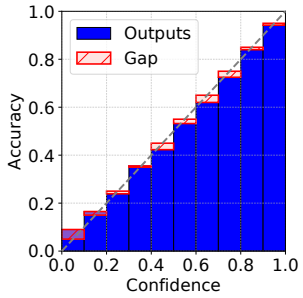
To this end, we assess model calibration, where a model's predictive confidence should align with its empirical accuracy.

Model	Continuum Dropout	CIFAR-100	CIFAR-10	STL-10	SVHN
Neural ODE	X	0.745 (0.006)	0.739 (0.008)	0.707 (0.007)	0.913 (0.004)
	O	**0.762 (0.002)****	**0.765 (0.007)****	**0.719 (0.005)****	**0.925 (0.004)****
Neural SDE (additive)	X	0.749 (0.003)	0.745 (0.009)	0.707 (0.004)	0.918 (0.003)
	O	**0.770 (0.004)****	**0.771 (0.006)****	**0.718 (0.002)****	**0.925 (0.003)****
Neural SDE (multiplicative)	X	0.753 (0.003)	0.747 (0.009)	0.705 (0.005)	0.914 (0.006)
	O	**0.768 (0.004)****	**0.768 (0.005)****	**0.715 (0.005)****	**0.924 (0.003)***

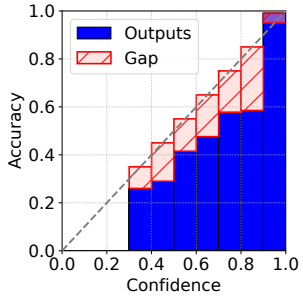
Table 6: Performance on four image datasets using Neural ODE and Neural SDEs



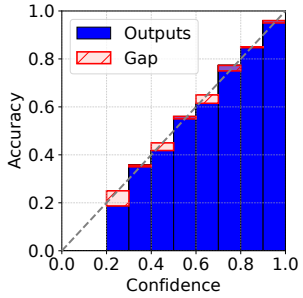
(a) CIFAR-100 without Continuum Dropout



(b) CIFAR-100 with Continuum Dropout



(c) Speech Commands without Continuum Dropout



(d) Speech Commands with Continuum Dropout

Figure 3: Reliability diagrams illustrating calibration performance on CIFAR-100 and Speech Commands datasets.

We use reliability diagrams to visualize this alignment; curves that lie close to the identity line indicate well-calibrated and thus more trustworthy uncertainty estimates. As shown in Figure 3, models with Continuum Dropout produce curves that more closely track the diagonal than their deterministic counterparts. This demonstrates that the injected stochasticity effectively regularizes overconfident predictions, leading to probability estimates that better reflect the model's true predictive uncertainty.

4.4 MC Sample Size vs. Performance and Cost

We conduct experiments to analyze the effect of Monte Carlo (MC) sample size during the test phase of Continuum Dropout, in terms of both model performance and computational cost. First, we apply the proposed method to models such as Neural CDE, ANCDE, Neural LSDE, Neural LNSDE, and Neural GSDE on the Speech Commands dataset, varying the number of MC samples to evaluate performance sensitivity. As shown in Figure 4, the accuracy of the models stabilizes once the number of MC samples exceeds 5. A similar trend is observed on the CIFAR-100 image dataset, and detailed results are provided in Appendix F.2.

Next, Table 7 reports the computation time in two settings: with and without Continuum Dropout. Although the number of MC sample increases the computational load of the proposed method, the overhead remains within an acceptable range, as MC simulations are performed only during the test phase and require inference on the trained model.

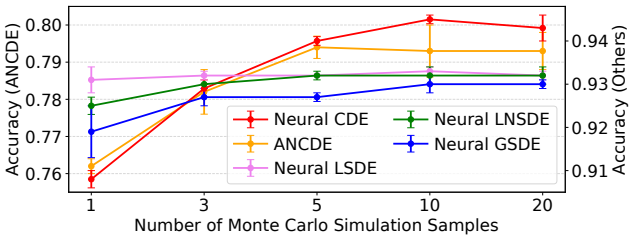


Figure 4: Performance with Different Numbers of MC Simulation Samples on Speech Commands

Continuum Dropout	Neural CDE	ANCDE	Neural LSDE	Neural LNSDE	Neural GSDE
X	25.6 (0.3)	53.3 (0.2)	19.4 (0.1)	19.5 (0.1)	19.8 (0.1)
O	27.1 (0.3)	58.9 (0.3)	21.6 (0.2)	21.7 (0.2)	22.5 (0.2)

Table 7: Computation time comparison on Speech Commands (time in seconds per epoch)

5 Conclusion

This paper introduced Continuum Dropout, a principled and universally applicable dropout framework for NDEs to enhance their generalization capabilities. Our core contribution is to formulate dropout for NDEs as an alternating renewal process, which provides a faithful continuous-time analogue of the discrete mechanism. We developed a novel, continuous-time definition for the dropout rate and a practical method to control it with intuitive hyperparameters. A key advantage of our framework is that it naturally enables uncertainty quantification via Monte Carlo sampling at test time. Extensive experiments demonstrated that Continuum Dropout consistently outperforms existing regularization methods and is universally applicable across a wide range of NDE architectures, achieving superior results on challenging time-series classification tasks. Furthermore, its ability to enhance model calibration highlights the effectiveness for uncertainty-aware modeling.

Acknowledgments

This research was supported by the Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No.RS-2020-II201336, Artificial Intelligence graduate school support(UNIST)); the Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. RS-2025-25442824, AI STAR Fellowship); the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. RS-2025-02216640); the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (RS-2024-00407852); the 2025 Research Fund (1.250006.01) of UNIST (Ulsan National Institute of Science & Technology); the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No.RS-2023-00218913).

References

Bagnall, A.; Dau, H. A.; Lines, J.; Flynn, M.; Large, J.; Bostrom, A.; Southam, P.; and Keogh, E. 2018. The UEA multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*.

Birolini, A. 1974. Some Applications of Regenerative Stochastic Processes to Reliability Theory - Part One: Tutorial Introduction. *IEEE Transactions on Reliability*, R-23(3): 186–194.

Chen, R. T. Q.; Rubanova, Y.; Bettencourt, J.; and Duvenaud, D. 2018. Neural Ordinary Differential Equations. *Advances in Neural Information Processing Systems*, 31.

Coates, A.; Ng, A.; and Lee, H. 2011. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 215–223.

Cox, D. R. 1962. *Renewal Theory*. Methuen.

Dau, H. A.; Bagnall, A.; Kamgar, K.; Yeh, C.-C. M.; Zhu, Y.; Gharghabi, S.; Ratanamahatana, C. A.; and Keogh, E. 2019. The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6): 1293–1305.

De Brouwer, E.; Simm, J.; Arany, A.; and Moreau, Y. 2019. GRU-ODE-Bayes: Continuous modeling of sporadically-observed time series. In *Advances in neural information processing systems*, volume 32.

Finlay, C.; Jacobsen, J.-H.; Nurbekyan, L.; and Oberman, A. M. 2020. How to Train Your Neural ODE: the World of Jacobian and Kinetic Regularization. In *International conference on machine learning*, 3154–3164. PMLR.

Gal, Y.; and Ghahramani, Z. 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *International Conference on Machine Learning*, volume 48, 1050–1059. PMLR.

Ghosh, A.; Behl, H. S.; Dupont, E.; Torr, P. H. S.; and Namboodiri, V. 2020. STEER: Simple Temporal Regularization For Neural ODEs. *Advances in Neural Information Processing Systems*, 33: 6696–6707.

Greydanus, S.; Dzamba, M.; and Yosinski, J. 2019. Hamiltonian Neural Networks. *Advances in Neural Information Processing Systems*, 32.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Heath, D.; Resnick, S.; and Samorodnitsky, G. 1998. Heavy Tails and Long Range Dependence in On/Off Processes and Associated Fluid Models. *Mathematics of Operations Research*, 23(1): 145–165.

Huang, X.; Ye, Y.; Xiong, L.; Lau, R. Y.; Jiang, N.; and Wang, S. 2016. Time series k-means: A new k-means type smooth subspace clustering for time series data. *Information Sciences*, 367: 1–13.

Jhin, S. Y.; Shin, H.; Kim, S.; Hong, S.; Jo, M.; Park, S.; Park, N.; Lee, S.; Maeng, H.; and Jeon, S. 2023. Attentive Neural Controlled Differential Equations for Time-series Classification and Forecasting. *Knowledge and Information Systems*, 1–31.

Kidger, P.; Morrill, J.; Foster, J.; and Lyons, T. 2020. Neural Controlled Differential Equations for Irregular Time Series. *Advances in Neural Information Processing Systems*, 33: 6696–6707.

Kong, L.; Sun, J.; and Zhang, C. 2020. SDE-Net: Equipping Deep Neural Networks with Uncertainty Estimates. In *International Conference on Machine Learning*, volume 37.

Krizhevsky, A.; and Hinton, G. 2009. Learning Multiple Layers of Features from Tiny Images.

Li, X.; Wong, T.-K. L.; Chen, R. T.; and Duvenaud, D. 2020. Scalable Gradients for Stochastic Differential Equations. In *International Conference on Artificial Intelligence and Statistics*, 3870–3882. PMLR.

Liaw, R.; Liang, E.; Nishihara, R.; Moritz, P.; Gonzalez, J. E.; and Stoica, I. 2018. Tune: A Research Platform for Distributed Model Selection and Training. *arXiv preprint arXiv:1807.05118*.

Liu, X.; Xiao, T.; Si, S.; Cao, Q.; Kumar, S.; and Hsieh, C.-J. 2019. Neural SDE: Stabilizing Neural ODE Networks with Stochastic Noise. *arXiv preprint arXiv:1906.02355*.

Liu, X.; Xiao, T.; Si, S.; Cao, Q.; Kumar, S.; and Hsieh, C.-J. 2020. How Does Noise Help Robustness? Explanation and Exploration under the Neural SDE Framework. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 279–287.

Löning, M.; Bagnall, A.; Ganesh, S.; Kazakov, V.; Lines, J.; and Király, F. J. 2019. sktime: A unified interface for machine learning with time series. *arXiv preprint arXiv:1909.07872*.

Moritz, P.; Nishihara, R.; Wang, S.; Tumanov, A.; Liaw, R.; Liang, E.; Elibol, M.; Yang, Z.; Paul, W.; Jordan, M. I.; et al. 2018. Ray: A distributed framework for emerging AI applications. In *13th USENIX Symposium on Operating Systems Design and Implementation OSDI 18*, 561–577.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading Digits in Natural Images with Unsupervised Feature Learning. *NIPS workshop on deep learning and unsupervised feature learning*, 2011(5): 7.

Oganesyan, V.; Volokhova, A.; and Vetrov, D. 2020. Stochasticity in Neural ODEs: An Empirical Study. *arXiv preprint arXiv:2002.09779*.

Oh, Y.; Kam, S.; Lee, J.; Lim, D.-Y.; Kim, S.; and Bui, A. A. T. 2025a. Comprehensive Review of Neural Differential Equations for Time Series Analysis. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, 10621–10631.

Oh, Y.; Lim, D.; and Kim, S. 2024. Stable Neural Stochastic Differential Equations in Analyzing Irregular Time Series Data. In *The Twelfth International Conference on Learning Representations*.

Oh, Y.; Lim, D.-Y.; and Kim, S. 2025. DualDynamics: Synergizing Implicit and Explicit Methods for Robust Irregular Time Series Analysis. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(18): 19730–19739.

Oh, Y.; Lim, D.-Y.; Kim, S.; and Bui, A. A. T. 2025b. TANDEM: Temporal Attention-guided Neural Differential Equations for Missingness in Time Series Classification. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, 1–11.

Pham-Gia, T.; and Turkkan, N. 1999. System Availability in a Gamma Alternating Renewal Process. *Naval Research Logistics (NRL)*, 46(7): 822–844.

Reyna, M. A.; Josef, C.; Seyedi, S.; Jeter, R.; Shashikumar, S.; Moody, B.; Westover, M. B.; Sharma, A.; Nemati, S.; and Clifford, G. D. 2019. Early Prediction of Sepsis from Clinical Data: The PhysioNet/Computing in Cardiology Challenge 2019. In *2019 Computing in Cardiology (CinC)*, 1–4. IEEE.

Ross, S. M. 1995. *Stochastic Processes*. Wiley.

Rubanova, Y.; Chen, R. T. Q.; and Duvenaud, D. 2019. Latent ODEs for Irregularly-Sampled Time Series. *Advances in Neural Information Processing Systems*, 32.

Sander, M.; Ablin, P.; and Peyré, G. 2022. Do Residual Neural Networks Discretize Neural Ordinary Differential Equations? *Advances in Neural Information Processing Systems*, 35: 36520–36532.

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal of Machine Learning Research*, 15: 1929–1958.

Stanford, R. E. 1979. Reneging Phenomena in Single Channel Queues. *Mathematics of Operations Research*, 4(2): 162–178.

Tzen, B.; and Raginsky, M. 2019. Neural Stochastic Differential Equations: Deep Latent Gaussian Models in the Diffusion Limit. *arXiv preprint arXiv:1905.09883*.

Wang, J.; Balasubramanian, A.; de la Vega, L. G. M.; Green, J. R.; Samal, A.; and Prabhakaran, B. 2013. Word Recognition from Continuous Articulatory Movement Time-Series Data using Symbolic Representations. In *Proceedings of the Fourth Workshop on Speech and Language Processing for Assistive Technologies*, 119–127.

Warden, P. 2018. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition. *arXiv preprint arXiv:1804.03209*.

Whitt, W. 2013. IEOR 3106: Alternating Renewal Processes and the Renewal Equation. Lecture notes, Department of Industrial Engineering and Operations Research, Columbia University.

Wilhelm, M.; Krakowczyk, D.; Trollmann, F.; and Albayrak, S. 2015. eRing: Multiple Finger Gesture Recognition with one Ring Using an Electric Field. In *Proceedings of the 2nd international Workshop on Sensor-based Activity Recognition and Interaction*, 1–6.

Yang, L.; Gao, T.; Lu, Y.; Duan, J.; and Liu, T. 2023. Neural network stochastic differential equation models with applications to financial data forecasting. *Applied Mathematical Modelling*, 115: 279–299.

Zheng, S.; Gao, Z.; Sun, F.-K.; Boning, D. S.; Yu, B.; and Wong, M. 2024. Improving Neural ODE Training with Temporal Adaptive Batch Normalization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.