The Journal of Supercomputing (2023) 79:18699–18727
https://doi.org/10.1007/s11227-023-05327-8

Time-aware neural ordinary differential equations for incomplete time series modeling

Zhuoqing Chang[1] · Shubo Liu[1] · Run Qiu[1] · Song Song[1] · Zhaohui Cai[1] · Guoqing Tu[2]

Accepted: 19 April 2023 / Published online: 18 May 2023

Abstract
Internet of Things realizes the ubiquitous connection of all things, generating countless time-tagged data called time series. However, real-world time series are often plagued with missing values on account of noise or malfunctioning sensors. Existing methods for modeling such incomplete time series typically involve preprocessing steps, such as deletion or missing data imputation using statistical learning or machine learning methods. Unfortunately, these methods unavoidable destroy time information and bring error accumulation to the subsequent model. To this end, this paper introduces a novel continuous neural network architecture, named Time-aware Neural-Ordinary Differential Equations (TN-ODE), for incomplete time data modeling. The proposed method not only supports imputation missing values at arbitrary time points, but also enables multi-step prediction at desired time points. Specifically, TN-ODE employs a time-aware Long Short-Term Memory as an encoder, which effectively learns the posterior distribution from partial observed data. Additionally, the derivative of latent states is parameterized with a fully connected network, thereby enabling continuous-time latent dynamics generation. The proposed TN-ODE model is evaluated on both real-world and synthetic incomplete time-series datasets by conducting data interpolation and extrapolation tasks as well as classification task. Extensive experiments show the TN-ODE model outperforms baseline methods in terms of Mean Square Error for imputation and prediction tasks, as well as accuracy in downstream classification task.

Keywords Incomplete time series · Time-aware encoder · Neural ODEs

✉ Shubo Liu
liu.shubo@whu.edu.cn

Extended author information available on the last page of the article

1 Introduction

Internet of Things (IoT) technology has matured to enable a wide spectrum of applications, spanning from environmental monitoring, smart transportation, smart grids to health care, food traceability and so on [1]. IoT devices generate massive amounts of heterogeneous data with time tags, which are named time series data. However, missing data is one of the commonly encountered problems in IoT time series analysis due to different sampling rates or unexpected failures in sensors or transmissions [2]. The missing items unavoidably obstruct the completeness of data, causing subsequent analysis to make wrong deductions [3].

Researchers have proposed varieties of counter measures to address this challenge. The simplest approach is to delete the missing data and to make inference only based on the observed data. It may work when missing data accounts for less than 15%, but significant barriers still remain [3]. This approach discards historical data to acquire complete information, and ignores much information hidden in the data. Additionally, it will result in data deviations and inaccurate predictions when the missing rate is high, especially when missing data are not randomly distributed. This technical issue may be addressed by the perspective of preprocess methods, namely missing data imputation, allowing full use of all available values. Generally, missing data can be estimated by statistical learning-based, machine learning-based and deep learning-based methods.

Some statistical methods have been adopted to impute missing data, such as mean value imputation, and zero value imputation [4]. Such methods are simple to implement, unfortunately, they do not consider the temporal relationship between variables, achieving unsatisfying imputation accuracy. Machine learning-based methods provide some useful insights on impute missing data. The *K*-nearest neighbor (KNN) [5], Autoregressive Integrated Moving Average models (ARIMA) [6], and the matrix factorization algorithm [7], representatives of machine learning methods, have been successfully used to impute missing data. However, they are trained in a manner, where models are trained using complete data and tested with incomplete data. Nonmissing values in the incomplete samples are excluded from the training process, unavoidably leading a degraded imputation performance. Generally, deep learning-based methods assume that time steps of time series are regular. Regarding this, several works [8] extend RNN to irregular time series by dividing the timespan into uniform intervals, and filling the missing values with averages. Such time discretization method inevitably destroys the measurement timing information, which can be informative about latent variables [9, 10]. Such two-stage methodology is still criticized for the difficulty of obtaining the complete data. Besides, data imputation, an independent step of data preprocessing, is separated from the training of data prediction, which will cause suboptimal results.

To combat this, some works directly model incomplete time series data by modifying architectures of deep learning models to learn time interval. A simple trick is to concatenate time information to the input of an RNN [9, 11]. Steps have been taken to incorporate time interval information into the model. GRU-D

employs a simple exponential decay between observations to modify the hidden state until the next observation is made [10]. BRITS conducts missing value imputation and classification/regression simultaneously using a bidirectional decay modified RNN [12]. Unfortunately, these approaches cannot predict the missing values of desired time point.

Recently, the Neural Ordinary Differential Equation (ODE) model starts to garner its share of the spotlight. Neural ODEs describe the input to output variable transformation by a continuous representation of trajectory through a vector field defined by a neural network [13]. The trajectory is generated by numerical ODE solver schemes, such as Euler's method or dopri5 method. As time of the trajectory is intrinsically continuous, the Neural ODE is considered as an ideal option for modelling temporal dynamics. There has been recent interest in modelling irregularly-sampled time series using Neural ODEs. A variational autoencoder (VAE)-based latent ODE is investigated for modeling irregularly-sampled time series by a continuous-time generation of latent trajectory, the underlying dynamics of which is determined by an initial latent state learned from partial observations using a RNN network. However, RNN models are not suitable for learning an approximated posterior distribution of observations that are not temporally aligned and faced with vanishing and exploding gradients. Additionally, RNN-ODE is put forwards that the hidden state is modelled using a Neural ODE and updated by a RNN network [14]. The hidden states learned by the Neural ODE in ODE-RNN encoder will cause the error accumulation in the following generative model.

This paper introduces a novel continuous neural network model for modeling incomplete time series. It is designed based on a VAE framework that involves neural ODEs to learn continuous-time latent dynamics. A time-aware LSTM network serves as the encoder which employs an extra time interval to decay the impact of previous observation on the current observation and a mask vector to indicate whether the data is missing or not. Incomplete time series is directly encoded by a time-aware encoder, which can effectively learn the temporal and sequential features and provide more adaptive input representations to the subsequent ODE network. The ODE parametered by a fully connected network can learn the relationship between unknown systems and their derivatives, enabling to impute or predict latent states at arbitrary steps. The proposed model outperforms state-of-art methods, demonstrating its great potential of modeling incomplete time series in real-world application. Ablation studies illustrate the effectiveness of time-aware component in the encoder of the proposed TN-ODE model in learning posterior distribution.

To conclude, the contributions of this paper can be summarized as follows:

- Proposing a novel TN-ODE model that can effectively handle the different time gaps between adjacent elements of the sequence dispensing with the additional data preprocessing.
- A time-aware LSTM network is used as an encoder to generate a more adaptive input to the ODE network, which is proved to be an effective parameterization when data are sparse and irregular.
- The TN-ODE model is tested on a synthetic and a true irregularly-sampled time series dataset and a UCI time series dataset with different missing rates by con-

ducting interpolation and extrapolation tasks, and evaluated on a real UCI irregularly-sampled IoT data via the human activity classification task. Results demonstrate that the proposed method shows substantially better performance.

The rest of this paper is organized as follows. Section 2 gives an overview of state-of-the-art methods for incomplete time series modeling. Section 3 presents problem definition and some preliminaries. Section 4 details the proposed TN-ODE framework. Section 5 elaborates on the proposed model. Finally, Sect. 6 concludes this paper.

2 Related work

This section lists the related research on incomplete time series data modeling, which can be roughly classified into two categories: two-stage methods and one-stage methods.

2.1 Two-stage methods

One approach for modeling incomplete time series is to use a two-stage method, whereby missing values are first imputed, followed by the application of existing classification or prediction methods to the completed data. Statistical methods, such as mean filling [4], fuzzy-rough nearest neighbors [15], have been used to fill in missing values. However, these methods are not suitable for missing data imputation in the context of IoT, which lack the usage of temporal information. Advancements in machine learning methods promote their applications of data filling techniques that concentrate on learning the distribution of original data and fitting a filling model. The KNN can be used to impute missing data by filling in the missing values with the average of$k$ neighbor nodes near the missing sample [5]. Matrix factorization is another technique that can be applied to impute the missing values [7]. However, traditional machine learning methods do not consider the temporal relations between observations, making them unsuitable for IoT big data environments [16].

The advent of Generative Adversarial Networks (GANs) opens up a new approach to imputing missing data values, such as GAIN [17], E2GAN [18] and so on. An Inverse Mapping General Adversarial Network named IM-GAN is designed based on a GAN structure to handle missing indoor air quality data [19]. In IM-GAN, a denoising auto-encoder is used as a generator to learn robust representation, the encoder of which employs a Bi-directional Recurrent Neural Network (BRNN) cell to model bi-directional temporal correlations and across-sensor correlations.E2GAN is proposed to impute missing values of clinical and meteorologic data using a new complete sample generated by auto-encoder and GRUI [18]. These models have achieved a great success in imputation task; however, they are typically used as a data pre-processing step for classification and prediction tasks. Furthermore, they are not optimized in concert with training the following networks, which may result in suboptimal results [20].

2.2 One-stage methods

One-stage methods utilize irregular time information to directly model raw time series data using improved deep learning methods. Several approaches modify the inputs of RNNs by concatenating missing entries or timestamps with the input [21, 22]. Other works aim to reconstruct RNN structures that incorporates time interval to the hidden units to handle missing data in real-world clinical settings [10, 23]. BRITS [24] is proposed for imputing missing values of multivariate time series with bidirectional recurrent dynamics, using delayed gradients for missing values in both forward and backward directions to improve imputation accuracy. A VAE-based encoder-decoder model leveraging a Multi-Time Attention (mTAN) module is presented, which can conduct interpolation and classification tasks on sparse and irregularly sampled data [25]. Regretfully, these models are capable of inferring missing data and performing regression/classification tasks simultaneously but cannot make prediction. A model named Time Encoding-Encoding Echo State Network (TE-ESN) is designed to support early prediction and one-step-ahead forecasting on irregularly sampled time series [26]. However, these models cannot predict multi-step data values of any desired time.

Recently, a continuous version of neural networks called Neural ODEs, is proposed by [13] to overcome the limitations imposed by discrete-time recurrent neural networks. The successful combination of Neural ODEs and a VAE paves a natural way to handle incomplete time series. It still remains a barrier that the discrete-time RNN encoder is hard to learn the posterior distribution from sporadic data. To address this issue, a new VAE architecture is designed that uses Neural ODEs model for encoding the data and generating the latent states [14]. However, the ODE-RNN encoder generates new data, leading to error accumulation when inferring the posterior distribution. Moreover, more extended Neural ODEs architectures are investigated to remedy the shortcomings of Neural ODEs. Augmented Neural ODEs are designed to mitigate the issue that Neural ODEs only learning features which are homeomorphic to the input space [27]. These methods are capable of forecasting the data at any desired time; however, encoders in these VAE-based ODE methods are unable to learn uneven time intervals. To reduce training time, the continuous recurrent units model is designed based on an encoder-decoder framework, hidden state of which evolve according to a linear stochastic differential equation [28].

Graphs data structures are widely used in various applications, such as social networks, recommendation systems, and traffic forecast systems. However, real-world graph data is often incomplete due to various factors, such as different sampling frequencies of different nodes and missing nodes. Great effort is put into developing effective techniques to model incomplete graph data. For instance, an adaptive graph recurrent network has been proposed, which combines graph and RNNs for air quality and traffic data imputation [29]. To model temporal session data in discrete state spaces, a graph nested GRU ODE model is proposed to preserve the continuous nature of dynamic user preferences, where a graph gated neural network is employed to encode both temporal and structural patterns for inferring initial latent states and a time alignment algorithm is designed to align the updating time steps of temporal session graphs [30]. For irregularly sampled and multivariate time series,

RAINDROP represents every sample as a separate sensor graph and captures time-varying dependencies between sensors with a novel message passing operator [31]. Additionally, to model a dynamical system like the COVID-19 pandemic, a coupled graph ODE model is established, which learns the coupled dynamics of nodes and edges with a graph neural network-based ODE in a continuous manner [32]. Moreover, a latent ODE generative model is presented to model multi-agent dynamic system with a known graph structure, capable of learning high-dimensional trajectory embeddings and inferring continuous latent system dynamics simultaneously [33].

3 Problem statement and preliminaries

This section begins by defining the problem formulation and then provides some background information on LSTM, VAEs, and Neural ODEs and then presents some background knowledge of LSTM, VAEs and Neural ODEs.

3.1 Problem definition

Let $X = \{x_1, x_2, \dots, x_N\} \in \mathbb{R}^{D \times N}$ be a D-dimensional time series consisting of N observations, where each observation $x_t \in \mathbb{R}^D$ is composed of D variables $\{x_t^1, x_t^2, \dots, x_t^D\}$ and is sampled at timestamp s_t. It is important to note that time intervals between timestamps may vary in duration. In real-world scenarios, unforeseen events such as sensor failures or communication errors may result in missing data in x_t. To identify the missing data in x_t, this paper introduces a mask matrix $M = \{m_1, m_2, \dots, m_N\} \in \{1, 0\} \subset \mathbb{R}^{D \times N}$, which indicates whether each observation is observed or missing, defined as follows:

$$m_t^d = \begin{cases} 1, & \text{if } x_t^d \text{ is observed} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

δ_t^d is defined as the time gap between the current timestamp s_t and the previous observation s_{t-1}, and is used to record this time gap.

$$\delta_t^d = \begin{cases} s_t - s_{t-1} + \delta_{t-1}^d, & \text{if } t > 1, m_{t-1}^d = 0 \\ s_t - s_{t-1}, & \text{if } t > 1, m_{t-1}^d = 1 \\ 0, & t = 1 \end{cases} \tag{2}$$

3.2 Long short-term memory

RNNs suffer from the issue of vanishing and exploding gradients, which hampers learning of long time series. The emergence of LSTMs can alleviate this problem. Given a sequence of inputs $X = \{x_1, x_2, \dots, x_N\}$, LSTMs model their cell memory and hidden states as a pair (c_t, h_t) and update the pair of hidden states according to the following recurrent equations.

$$f_t = \sigma(v_f x_i^t + u_f h_i^{t-1} + b_f) \tag{3}$$

$$i_t = \sigma(v_i x_i^t + u_i h_i^{t-1} + b_i) \tag{4}$$

$$o_t = \sigma(v_o x_i^t + u_o h_i^{t-1} + b_o) \tag{5}$$

$$\widetilde{c}_t = tanh(v_c x_i^t + u_c h_i^{t-1} + b_c) \tag{6}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \widetilde{c}_t \tag{7}$$

$$\begin{aligned} &h_t = o_t \circ tanh(c_t) \\ &\forall t \in \{1, 2, \dots, N\} \end{aligned} \tag{8}$$

where σ is the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$, the matrices v_p, u_p, and vectors b_p for $\forall p \in \{f, i, o, c\}$ are trainable parameters, $\circ$ means element-wise product. c_t and h_t are cell memory and hidden state, respectively. A LSTM cell employs four gates to manage its states over time aimed at alleviating the challenge of exploding/vanishing gradients when dealing with long time series. LSTMs are capable of dealing with equidistant streams of data, however, they are still unable learning time series data with varying time intervals. In order to tackle this, Sect. 4 will introduce an improved LSTM that incorporates the elapsed time to adjust the memory state.

3.3 Variational auto-encoder

The VAE is an important type of generative models, and this section will briefly summarize the principle of the VAE proposed by [34]. The original dataset $X = \{x_1, x_2, \dots, x_N\}$ consists of N continuous variable samples. It is assumed that the observations are depended on the unobserved random variable Z, which is considered to be a lower-dimensional latent space. A sample z_i is generated from a prior distribution $p_\theta(z)$ and x_i from a likelihood $p_\theta(x \mid z)$. The objective is to model or approximate the observations' true distribution $p_\theta(x)$ by maximizing the marginal likelihood $p_\theta(x) = \int p_\theta(x \mid z) p_\theta(z) \mathrm{d}z = \mathbb{E}[p_\theta(x \mid z)]$. However, when calculating the posterior distribution, since z is unknown, the posterior distribution cannot be obtained directly. To handle this, the application of variational inference (VI) is introduced. Suppose that there exists a probability distribution $q_\phi(z \mid x)$ that is sufficiently close to $p_\theta(x \mid z)$ and can be calculated. The issue of calculating the posterior distribution is converted to solving θ and ϕ that make $q_\phi(z \mid x)$ and $p_\theta(x \mid z)$ sufficiently approximated.

Closeness of the two distributions is quantified based on the Kullback-Leibler (KL) divergence D_{KL}, which is utilized as a measure for information when utilizing a distribution to represent another intractable distribution [35]. For the KL divergence of $q_\phi(z \mid x)$ and $p_\theta(x \mid z)$ we have that

$$
\begin{aligned}
D_{KL}(q_\phi(z \mid x) \mid\mid p_\theta(z \mid x)) &= \int q_\phi(z \mid x)\log\frac{q_\phi(z \mid x)}{p_\theta(z \mid x)}\mathrm{d}z \\
&= \int q_\phi(z \mid x)\log\frac{q_\phi(z \mid x)p_\theta(x)}{p_\theta(z,x)}\mathrm{d}z \\
&= \int q_\phi(z \mid x)(\log p_\theta(x) + \log\frac{q_\phi(z \mid x)}{p_\theta(z,x)})\mathrm{d}z \\
&= \log p_\theta(x) + \int q_\phi(z \mid x)\log\frac{q_\phi(z \mid x)}{p_\theta(z,x)}\mathrm{d}z \\
&= \log p_\theta(x) + \int q_\phi(z \mid x)\log\frac{q_\phi(z \mid x)}{p_\theta(x \mid z)p_\theta(z)}\mathrm{d}z \\
&= \log p_\theta(x) + \mathbb{E}_{z\sim q_\phi(z|x)}[\log\frac{q_\phi(z \mid x)}{p_\theta(z)} - \log p_\theta(x \mid z)] \\
&= \log p_\theta(x) + D_{KL}(q_\phi(z \mid x) \mid\mid p_\theta(z)) - \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x \mid z)]
\end{aligned}
\tag{9}
$$

where $\mathbb{E}$ is the expectation operator. Rearrangement yields

$$
\begin{aligned}
&\log p_\theta(x) - D_{KL}(q_\phi(z \mid x) \mid\mid p_\theta(z \mid x)) \\
&\quad = \mathbb{E}_{z\sim q_\phi(z|x)}(\log p_\theta(x \mid z)) - D_{KL}(q_\phi(z \mid x) \mid\mid p_\theta(z))
\end{aligned}
\tag{10}
$$

The left-hand side of equation Eq. (10) is what we desire to maximize. The log-likelihood of generating true data $\log p_\theta(x)$ supposes to be higher and the difference between estimated and true posterior distributions $D_{KL}(q_\phi(z \mid x) \mid\mid p_\theta(z \mid x))$ should be as low as possible. Considering the non-negative property of $D_{KL}(q_\phi(z \mid x) \mid\mid p_\theta(z \mid x))$, it can be asserted that $\log p_\theta(x) \geq \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x \mid z)] - D_{KL}(q_\phi(z \mid x) \mid\mid p_\theta(z))$, and the right-hand side of Eq. (10) is named the Evidence Lower Bound (ELBO) for $\log p_\theta(x)$.

The loss function $L_{VAE}(\theta, \phi)$ consists of two parts, one is the reconstruction error $\mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x \mid z)]$, and the other part is the KL penalty $D_{KL}(q_\phi(z \mid x) \mid\mid p_\theta(z))$, which is used to constrain the encoder to approximate the prior distribution $p(z)$. The total loss can be expressed as follows.

$$
L_{VAE}(\theta, \phi) = -\mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x \mid z)] + D_{KL}(q_\phi(z \mid x) \mid\mid p_\theta(z)) \tag{11}
$$

Figure 1 presents a graphical representation of the VAE, which consists of an encoder and decoder. $q_\phi(z \mid x)$ is the encoder model, which is used to mean and variance of sampled data, obtain the corresponding normal distribution and extract features of low-dimensional latent states z. $p_\theta(x \mid z)$ is the decoder model, which is responsible for generating a reconstructed data.

3.4 Neural ordinary differential equations

Neural ODEs, proposed by [13], are interpreted as a continuous version of the residual neural networks (ResNets) modelled by differential equations, which can break

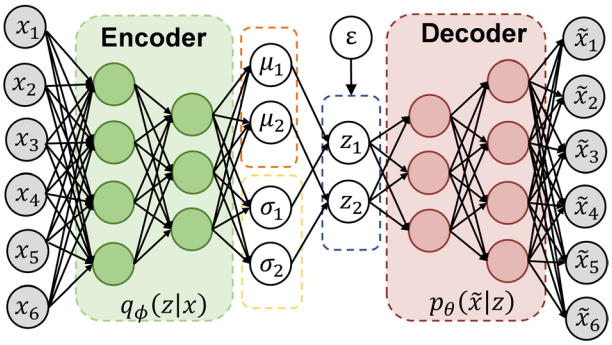

Fig. 1 The structure of the VAE model

through the bottlenecks imposed by discrete-time RNNs. The residual layer updates its hidden state at time i based on a transformation f over the previous state, written as $h_i = h_{i-1} + f(h_{i-1})$. Unlike this discrete update, the equation describing Neural ODEs is denoted as follows.

$$h(\tau) = h(0) + \int_0^\tau f_\theta(h(t), t; \theta_f)\mathrm{d}t \tag{12}$$

where the derivative of the hidden state $\frac{h(t)}{\mathrm{d}t}$ is approximated using a neural network parameterized by f_θ. In this way, the hidden state $h(t)$ is defined as a solution to ODE initial-value problem. The hidden states at any desired time can be obtained by solving the integral problem via various numerical ODE solvers (such as Euler method, Dormand-Prince method and so on), simplified as.

$$h_0, \ldots, h_n = ODESolve(f_\theta, h_0, (t_0, \ldots, t_n)) \tag{13}$$

It is worth mentioning that Eq. (12) can be regarded as the residual connection when the explicit Euler method is chosen to solve an ODE. Inversely, Neural ODEs generalize ResNets with a continuous time variable t. Additionally, the adjoint sensitivity method [36] is employed to compute memory efficient gradients w.r.t. θ for training Neural ODE-based models using black-box ODE solvers in [13].

4 Proposed methods

In this section, the proposed TN-ODE algorithm will be introduced in detail. Section 4.1 gives an overview of the framework, which consists of three components, namely the time-LSTM encoder, the generative model of latent states and the decoder. The time-aware LSTM encoder will be described in Sect. 4.2. In Sect. 4.3, the generative model of latent states will be discussed. Additionally, Sect. 4.4 will present the decoder. Finally, loss functions are envisioned for effectively training the proposed TN-ODE algorithm and the variants of RNN.

4.1 Overview of the TN-ODE algorithm

As illustrated in Fig. 2, the TN-ODE algorithm is based on the VAE model, which includes a time-aware LSTM encoder, a generative model and a decoder. The encoder learns the latent initial state from observed data, however, the observed data are usually confronted with missing values or uniform time intervals, which are an ill fit for traditional deep learning model. Thus, a time-aware LSTM encoder is employed to directly infer the latent initial state z_0 from partially-observed trajectories. The latent dynamics z_{t_i} are inferred by a generative model defined by an ODE function based on the sampled initial states. A fully connected network serves as the decoder which can recover the trajectory according to decoding likelihood $p(o_{t_i} \mid z_{t_i})$. The time-aware LSTM encoder receives the IoT time series data $X = \{x_1, x_2, \ldots, x_N\}$ in a chronological order and obtain the hidden states $\{h_1, h_2, \ldots, h_N\}$. The local initial state z_0 is learned based on the finally hidden state h_N. Given the desired times and the initial state z_0, the latent ODE generates a latent trajectory, which represents the latent states at each time step. The value of any desired time can be inferred by decoding the latent states. In the following, the three components will be described in detail.

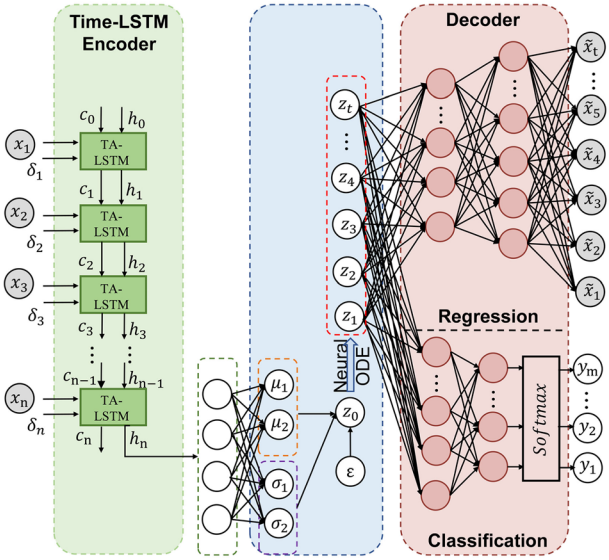

Fig. 2 The proposed TN-ODE framework

Algorithm 1 The proposed model

Require: Data points with their timestamps $\{x_i, t_i\}_{i=0,1,\ldots,N}$,
Initial hidden state h_0, Initial memory cell c_0
Ensure: A predicted time series $\widetilde{x}_{1:T}$

```
h_0 ⇐ 0, c_0 ⇐ 0
if i < N then
    Compute h_i, c_i = f_enc{x_i, t_i}_(i=0,1,...,N)
end if
Draw N(μ_z, σ_z) = g(h_{N-1})
Draw z_0 = μ_z + σ_z ∘ ε
Compute {z_1, z_2, ..., z_T} = ODESolve(z_0, f_θ, {t_0, t_{1:T}})
Compute {x̃_1, ..., x̃_T} = f_dec({z_1, z_2, ..., z_T})
```

4.2 The time-aware LSTM encoder

Real-world time series data often suffer from issues of missing values and irregular time intervals, which has a disastrous influence on deep learning models. Time-aware LSTM is proposed by [23] to address the nonuniform time lapse between successive elements. Thus, in this paper, a time aware LSTM encoder is introduced to infer latent initial states from partially-observed data. Time-aware LSTM incorporates the elapsed time between consecutive observations into basic LSTM by a time decay function $\beta(\delta_i^t) = \frac{1}{\log(e+\delta_i^t)}$, which is validated by previous works [23, 37] using irregularly sampled clinical time series. The time-aware LSTM performs a subspace decomposition of the previous memory cell C_{t-1}, forming a short-term memory C_{t-1}^S and a long-term memory C_{t-1}^L. The short-term memory is obtained via a network and is discounted via a decay function of elapsed time to capture the irregular temporal dynamics, yielding the discounted short-term memory $\widehat{C}_{t-1}^S$. After that, the long-term memory ($C_{t-1}^L = C_{t-1} - C_{t-1}^S$) is calculated. Finally, an adjusted previous memory $C_{t-1}^{'}$ is established by adding the long-term memory and the discounted short-term memory. The adjusted previous memory along with h_{t-1} and x are further calculated as in LSTM by substituting C_{t-1} with $C_{t-1}^{'}$. Detailed mathematical computations of time-aware LSTM are summarized below:

$$C_{t-1}^S = tanh(\omega_s C_{t-1} + b_s) \tag{14}$$

$$\widehat{C}_{t-1}^S = C_{t-1}^S \times \beta(\delta_i^t) \tag{15}$$

$$C_{t-1}^L = C_{t-1} - C_{t-1}^S \tag{16}$$

$$C'_{t-1} = C^L_{t-1} + \widehat{C}^S_{t-1} \tag{17}$$

$$f_t = \sigma(v_f x^t_i + u_f h^{t-1}_i + b_f) \tag{18}$$

$$i_t = \sigma(v_i x^t_i + u_i h^{t-1}_i + b_i) \tag{19}$$

$$o_t = \sigma(v_o x^t_i + u_o h^{t-1}_i + b_o) \tag{20}$$

$$\widetilde{C} = tanh(v_c x^t_i + u_c h^{t-1}_i + b_c) \tag{21}$$

$$C_t = f_t * C'_{t-1} + i_t * \widetilde{C} \tag{22}$$

$$h_t = o_t \circ tanh(C_t) \tag{23}$$

where the matrices v_p, u_p, and vectors b_p for $p \in \{f, i, o, c\}$ are the parameters to be trained for constructing the time-aware LSTM network. Time-aware LSTM is used for probabilistic encoder $q_\phi(z \mid x)$, which is set to be a multivariate Normal distribution

$$\log q_\phi(z \mid x) = \log N(z; \mu_z, \sigma^2_z) \tag{24}$$

where μ_z is the mean and σ^2_z is the covariance matrix.

4.3 The generative model of neural ODE

The last hidden state of the time-aware LSTM encoder is converted to (μ_z, σ^2_z) by a simple neural network. z is obtained by sampling from $q_\phi(z \mid x)$

$$z \sim q_\phi(z \mid x) \tag{25}$$

However, the stochastic sampling is a non-differentiable operation that do not support backpropagating the gradient during training. The reparameterization trick is employed to modify Eq. (25), where z can be represented as the sum of a deterministic variable and an auxiliary independent random variable ε.

$$z = \mu_z + \sigma_z \circ \varepsilon \tag{26}$$

where $\circ$ defines the element-wise product and $\varepsilon \sim N(0, I)$.

To construct continuous-time latent dynamics, a Neural ODE is adopted, the dynamics of which is entirely determined by the initial state z_0 sampled from the above encoding process. According to the initial state z_0, an ODE solver produces a series of latent states z_{t_i} of arbitrary desired time steps t_i on a continuous timeline.

$$z_1, z_2, \ldots, z_T = ODESolve(z_0, f_\theta, \{t_0, t_{1:T}\}) \tag{27}$$

where f_θ is a time-invariant function that defines the gradient $\partial z(t)/\partial t$, which is parametrized by a fully connected neural network.

4.4 The fully connected network and softmax decoder

The decoder $p_\theta(\tilde{x} \mid z)$ targets to reconstruct X using the series of latent states Z generated by the Neural ODE. This paper utilizes two different decoders according to the regression and classification tasks. A linear network is employed as the decoder to perform interpolation and extrapolation tasks, which finally outputs $\widetilde{X}$ based on the latent trajectory. For the task of classification, a *softmax* activation layer is utilized to estimate the probability distribution $\widetilde{Y}$ of activity at each time point.

4.5 Training the TN-ODE model

Unsuperivsed learning To learn the parameters of the TN-ODE model using a dataset of incomplete time series, this paper follows a VAE method. The learning objective is defined as follows, where all the formulas are defined in previous section. Two losses are introduced in the regression tasks. The first loss is reconstruction error, which sums the square error between the input X and output $\widetilde{X}$. The reconstruction error loss $loss_{Error}$ is calculated by summing the error over both the feature (D features) and the time dimension (N time points), presented below.

$$loss_{Error} = \sum_{t=0}^{N-1} \sum_{i=0}^{D-1} (x_i^t - \tilde{x}_i^t)^2 \tag{28}$$

The second loss is KL divergence between prior distribution $p_\theta(z)$ and posterior distribution $q_\phi(z \mid x)$, aimed at making them closer in latent space:

$$loss_{KL} = \frac{1}{M} \sum_{i=1}^{M} D_{KL}(q_\phi(z \mid x) \,||\, p_\theta(z)) \tag{29}$$

where D_{KL} is defined in Eq. (9). Therefore, the total loss for regression is defined as follows. And the adjoint backpropagation method is used to save more memory and computing resources.

$$loss_{Regression} = loss_{Error} + loss_{KL} \tag{30}$$

Supervised learning The TN-ODE model is also trained by a supervised learning approach. The time series classification task can be viewed as an illustrative supervised learning problem. The latent states are followed by a *softmax* module to conduct the classification model $p_\theta(\tilde{y} \mid z_t)$. The standard cross-entropy loss $loss_{Classification}$ is used with latent state vector z_t, followed by a classification layer as follows:

$$\tilde{y} = softmax(w \cdot z_t + b) \tag{31}$$

$$loss_{Classification} = -\frac{1}{N}\sum_{i=0}^{N-1}\sum_{j=0}^{M-1} y_{i,j} ln\tilde{y}_{i,j} \tag{32}$$

where N and M denote the number of data points and labels and $y_{i,j}$ represents that the true label of the ith data point is k and $\tilde{y}_{i,j}$ is the softmax probability for the ith data point.

5 Experimental details

In this section, experiments have been carried out to evaluate the performance of the TN-ODE model on two public benchmark datasets along with a sine wave simulation. Several well-established models are run as the baseline methods, including GRU, GRU-D, T-LSTM, BRITS, AJ-RNN, models based on VAE coupled with Neural ODEs and different encoder variants, such as RNN, GRU and ODE-RNN encoder.

5.1 Experiment settings

The experiment setting including dataset description and evaluation metrics are first introduced.

5.1.1 Dataset description

Sine wave simulation The sine wave simulation of 1000 periodic trajectories with the same frequency and amplitude is synthesized according to the research [13]. All the initial points are sampled from a standard Gaussian, and the Gaussian noise are injected into the observations. 100 time points are selected at random in each trajectory. For training, a full set are constructed based on the 100 time points.

PhysioNet challenge 2012 dataset The healthcare dataset consists of 8000 clinical time series with measurements from the first 48 h of each individual's admission to intensive care unit (ICU) [38]. The measurements are sampled at irregular times, and of varying sparse subsets of 37 possible features.

Electricity The raw dataset of electricity is downloaded from the web [39]. The electricity consumption in kWh is collected every 15 min during the year from 2012 to 2014 for 321 clients. The data is converted to represent hourly consumption.

Human activity dataset The classifier model is trained on the human activity dataset from the UCI repository [40], which aimed to recognize the seven types of activities (walking, sitting, etc.) of five individuals based on data sensed by sensors worn on the person's belt, chest and ankles (12 features in total). The data is recorded at a fixed period of 211 ms, however, the random phase-shifts between them makes an irregularly sampled time-series.

5.1.2 Evaluation metrics

In this paper, Mean Square Error (MSE), Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) are used as the evaluation metrics for interpolation and extrapolation tasks, which are formulated with Eqs. (33), (34) and (35), respectively.

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y^i - \tilde{y}^i)^2 \tag{33}$$

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y^i - \tilde{y}^i)^2} \tag{34}$$

$$MAE = \frac{1}{m}\sum_{i=1}^{m} \mid y^i - \tilde{y}^i \mid \tag{35}$$

Accuracy is the most intuitive evaluation metric and reflects the ratio of correctly predicted observation to the total observations. It is used to evaluate the performance of the classification task, as defined below:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{36}$$

where TP, TN, FP and FN refer to the numbers of true positives, true negatives, false positives, and false negatives, respectively.

5.1.3 Setting of this experiment

The TN-ODE model and baseline models are all developed using the Pytorch 1.0. The ODE generative model is implemented with a standard ODE solver torchdiffeq in Python. The Adamax optimization method is employed to train all the models and learning rate of all the models is set to 0.001. Experiments are performed on a regular workstation equipped with a single NVIDIA Quadro P6000 graphics card of 24 GB memory and Ubuntu 18.04 system. To ensure a fair comparison, all the training parameters are tuned for best performance.

5.2 Baselines

The TN-ODE model is compared with the following baselines.

GRU [41]: Gated Recurrent Unit. It is an advancement of the standard RNN that can capture long-term temporal dependencies.

GRU-D [10]: It is a GRU-based model that introduces data and corresponding time steps into standard GRU to deal with multivariate time series prediction with missing values/irregular samplings.

T-LSTM [23]: Time-aware LSTM (T-LSTM) incorporates the irregular time intervals into the standard LSTM architecture to adjust the hidden status in the memory cell, aimed at handling sequential data with time irregularities.

BRITS [12]: A novel method directly learns the missing values in a bidirectional recurrent dynamical system, without any specific assumption over the data.

AJ-RNN [20]: This end-to-end model is trained in an adversarial and joint learning manner where the generator imputes the missing values and the discriminator to improve the imputation, which makes imputation more accurate.

Latent ODE [13]: It is a generative model based on VAE, where the Neural ODE generates the latent states at desired times according to initial state learned from the RNN encoder and a linear layer decodes the latent trajectory to impute the missing values.

GRU-VAE: It is an enhanced Latent ODE model which employs an GRU model as encoder to avoid the exploding gradients.

ODERNN-VAE [14]: It is an improved version of Latent ODE, where ODE-RNN serves as an encoder to provide a better approximate posterior than RNN on sparse data.

5.3 Imputation and prediction performance

A number of experiments are carried out to evaluate the performance of variants of both RNN and proposed approaches on the imputation and prediction tasks.

5.3.1 Imputation and prediction performance of irregular time series

The Sine Wave Simulation is generated by a sine function of the form $A\sin(\omega t) + G(d)$, where the amplitude A is set to 1 and the frequency ω is set to π. Here, $G(d)$ denotes Gaussian random noises. To generate irregular time stamps of a sequence, 100 time steps are randomly selected from a range of 0 to 5. And 1000 sequences with different time points are created, among which 800 sequences are used for validation and 200 for testing.

To demonstrate the ability of the TN-ODE model of inferring data values for any desired time steps, the autoregressive task is conducted on the irregular sine wave simulation with 1D sequential data. To verify its effectiveness, the TN-ODE model is conditioned on the subset of irregular data points from 0 to 5 and reconstruct the full trajectory in the same time range, which can be view as an interpolation task. As shown in Fig. 3, the red dot represents the observed irregular time series and the blue line represents the recovery trajectory. Additionally, the model is trained to condition on 50 points ranging from 0 to 2.5 (yellow dot in Fig. 3) and predict trajectory from 2.5 to 5 (green line in Fig. 3), which is considered as an extrapolation task. Despite being conditioned on irregular data, the TN-ODE model is able to

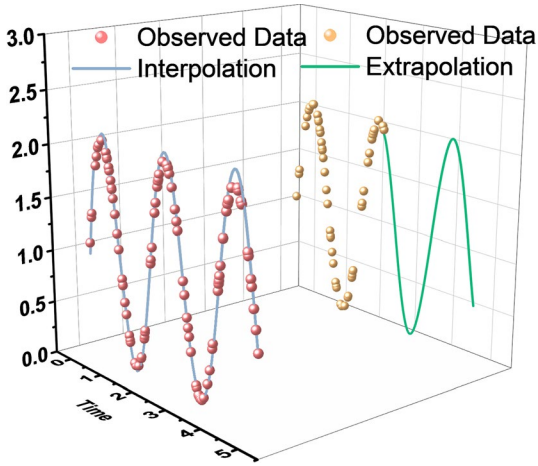

Fig. 3 Interpolation and extrapolation of sine wave simulation with irregular time steps by the TN-ODE model

reasonably reconstruct trajectories and provide an estimate of uncertainty for predicted observations.

To quantitatively analyze the performance of the TN-ODE model on the interpolation task, MSE, RMSE and MAE are reported on 100 time points that are used for training. These metrics are also compared with various comparison methods. RNNs cannot handle the long-distance dependency issue and suffer from a gradient vanishing problem, thus this paper selects an improved version of RNN, such as GRU, as baseline algorithms. The Latent ODE model has two defects: the encoder cannot learn irregular time intervals effectively, and RNNs have a vanishing gradient problem. To address the shortcomings, this paper employs a time-aware LSTM. Table 1 presents performance of the TN-ODE model and baseline algorithms. As can be seen, the TN-ODE model outperforms traditional methods including GRU, GRU-D, T-LSTM, BRITS, AJ-RNN and recent VAE-based ODE methods with different encoders (i.e. RNN encoder, GRU encoder and ODE-RNN encoder). It

Table 1 Interpolation results on the Sine Wave Simulation

Method	Inter		
	MSE	RMSE	MAE
GRU	0.0264	0.1625	0.1175
GRU-D	0.0263	0.1622	0.1186
T-LSTM	0.0327	0.1809	0.1224
BRITS	0.0119	0.1090	0.0578
AJ-RNN	0.0173	0.1314	0.0930
Latent ODE	0.0550	0.2346	0.1654
GRU-VAE	0.1156	0.3400	0.2390
ODERNN-VAE	0.0110	0.1050	0.0747
TN-ODE (ours)	0.0106	0.1027	0.0729

is obviously observed that the proposed TN-ODE significantly outperforms all baselines by achieving the lowest MSE, MAE and RMSE. The Neural ODE-based model performs better in the interpolation task than RNN-based models, possibly due to the fact that RNN-based models require explicit memory of historical information when processing long sequences, which makes them vulnerable to factors such as noise and missing data and face long-term dependency issues. Neural ODE-based models utilize a differential equation model to infer the relationship between time steps, allowing them to infer the dependency relationship between time steps without explicitly memorizing historical information. Thus, they have better representation learning ability, making them more robust to missing data and noise and easier to adapt to new data. Among the ODE-based method, the TN-ODE model has better results due to the ability of time-aware encoder to learn the time interval of time series and effectively infer the posterior distribution of irregularly-sampled time series data.

To further demonstrate the effectiveness of the proposed model in handling irregular time series, TN-ODE is evaluated on the Physionet Dataset by performing the interpolation task. Following the approach described in [14], observation times are rounded to the nearest minute to reduce the number of measurements by only twofold, leaving 2880 (60*48) possible measurement times per time series. Table 2 shows the comparative results of the imputation accuracy between TN-ODE method and other baseline methods. The proposed method outperforms the other imputation methods. Additionally, the Latent ODE also demonstrates good performance. Compared with GRU-D and T-LSTM, ODE-based methods adopt stronger differential equations to describe the dynamic behavior of the system, which enables them to learn more dynamic behaviors and better handle complex time series data. Moreover, the random sampling method used in ODE-based methods can sample the latent states of the system and generate continuous time dynamics, which helps to handle irregular time series data more effectively.

Table 3 presents the performance of TN-ODE model and baseline algorithms on the extrapolation task. The TN-ODE model outperforms the baseline algorithms in terms of predictive MSE, RMSE, and MAE. This is mainly because the TN-ODE

Table 2 Imputation performance of all methods on the Physionet Dataset in terms of MSE, RMSE and MAE

Method	Inter		
	MSE	RMSE	MAE
GRU	0.0079	0.0538	0.0467
GRU-D	0.0073	0.0511	0.0446
T-LSTM	0.0104	0.0627	0.0576
BRITS	0.1325	0.3640	0.2511
AJ-RNN	0.0109	0.0652	0.0600
Latent ODE	0.0070	0.0494	0.0448
GRU-VAE	0.0134	0.0693	0.0645
ODERNN-VAE	0.0081	0.0558	0.0509
TN-ODE (ours)	0.0059	0.0459	0.0412

Table 3 Extrapolation prediction performance on the Sine Wave Simulation

Method	Extrap		
	MSE	RMSE	MAE
Latent ODE	0.1521	0.3900	0.2772
GRU-VAE	0.4637	0.6801	0.6040
ODERNN-VAE	0.0227	0.1506	0.0930
TN-ODE (ours)	0.0021	0.0457	0.0317

model can effectively perceive the time interval changes of irregularly-sampled data and better learn the posterior distribution, resulting in a better ability to model the system dynamics. Therefore, this clearly illustrates that the time-aware LSTM encoder in the TN-ODE model overcomes the limitations of the RNN encoder used in Latent ODE.

The prediction task is also evaluated on the Physionet Dataset, and the experimental results are shown in Table 4. Consistent with the test results on Sine Wave Simulation, TN-ODE demonstrates the highest prediction accuracy by showing the lowest MSE. There is no doubt that the ODE module can better learn the system dynamics according to the posterior distribution learned by the time-aware encoder.

5.3.2 Imputation and prediction performance of time series with different missing rates

The Electricity Dataset is a widely-used public dataset from UCI, collected from 321 clients. For this experiment, data is selected from the period of 2012/01/01 to 2014/12/31, which has no missing data. Time-series samples for the interpolation and extrapolation tasks are chosen every 100 consecutive steps. The training and testing data are divided into 80% and 20% sets, respectively. To simulate missing values, various missing rates ranging from 10 to 70% are adopted to artificially drop observations in the training set and test set. And the observed values to eliminate are selected completely at random setting and independently for each time series. Average performance on the interpolation and extrapolation tasks under different missing rates is presented in Tables 5 and 6, respectively. The interpolation and extrapolation performance of 0–70% missing values are illustrated in Figs. 4 and 5.

As the MSE metric is shown in Fig. 4a, the traditional GRU, GRU-D, T-LSTM and BRITS demonstrate better data interpolation ability when the missing rate is less

Table 4 MSE, RMSE and MAE results of the TN-ODE and other extrapolation methods on the Physionet Dataset

Method	Extrap		
	MSE	RMSE	MAE
Latent ODE	0.0043	0.0316	0.0296
GRU-VAE	0.0062	0.0413	0.0362
ODERNN-VAE	0.0051	0.0366	0.0330
TN-ODE (ours)	0.0041	0.0308	0.0288

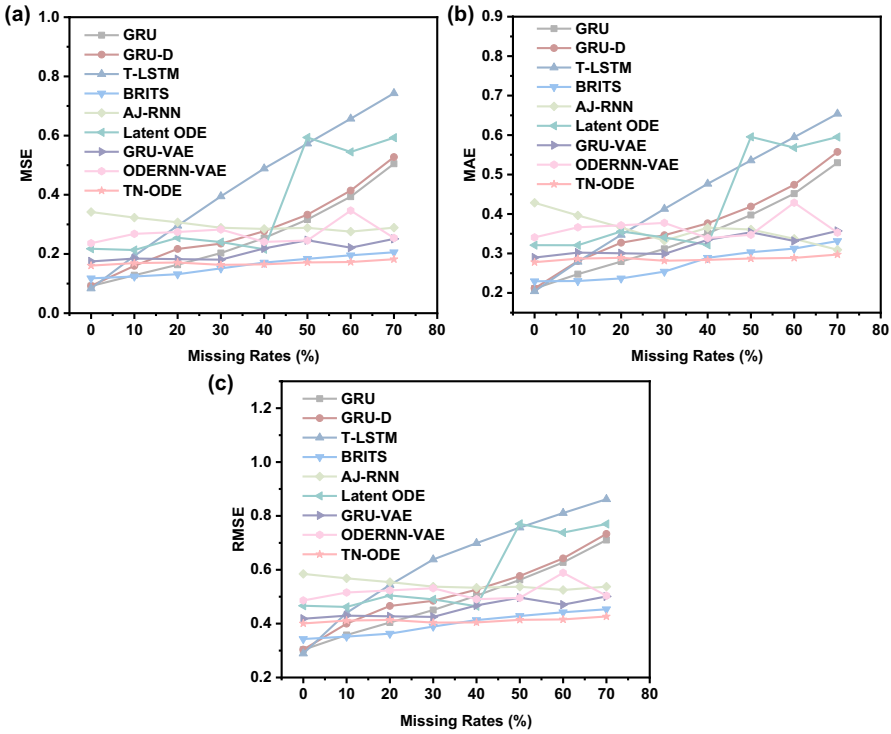

Fig. 4 **a** MSE, **b** RMSE and **c** MAE results of the Electricity Dataset under different missing rates in the interpolation task

Table 5 Average performance on the interpolation task using electricity test set under all missing rates

Method	Interp		
	MSE	RMSE	MAE
GRU	0.2569	0.4900	0.3475
GRU-D	0.2819	0.5166	0.3740
T-LSTM	0.4286	0.6299	0.4380
BRITS	0.1600	0.3980	0.2733
AJ-RNN	0.2998	0.4604	0.4200
Latent ODE	0.3593	0.5833	0.4273
GRU-VAE	0.2076	0.4545	0.3210
ODERNN-VAE	0.2686	0.5170	0.3655
TN-ODE (ours)	0.1694	0.4114	0.2865

than 15%. For missing data between 15 and 40%, only the BRITS algorithm outperforms TN-ODE. When the missing rate exceeds 50%, the TN-ODE model performs better than all baseline algorithms. Obviously, the interpolation performance of the RNN-based algorithms gradually declines as the data missing rate increases. This

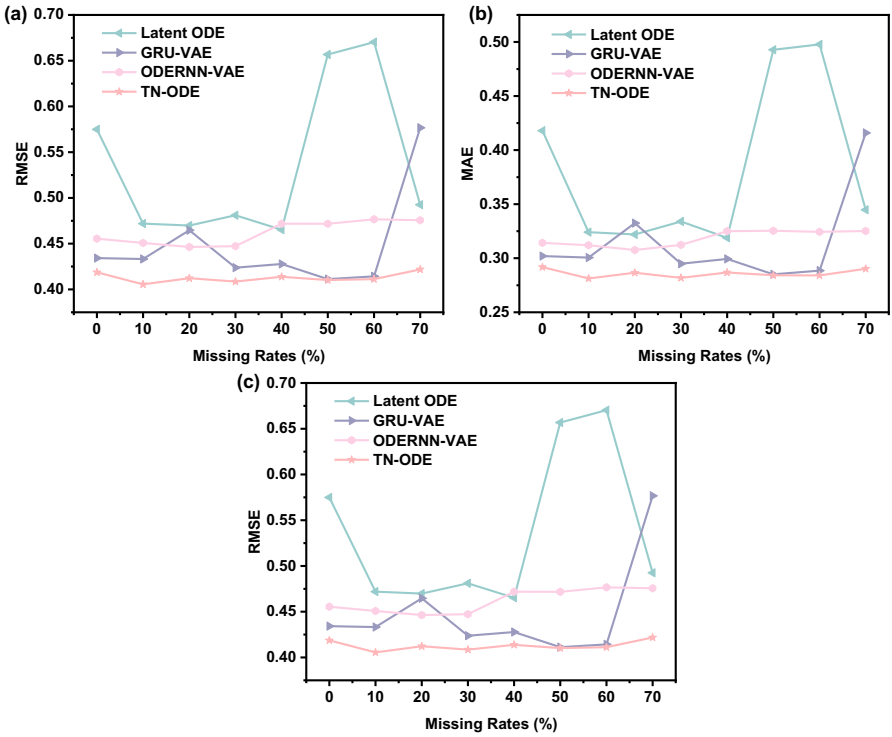

Fig. 5 **a** MSE, **b** RMSE and **c** MAE performance of the extrapolation task on the Electricity Dataset with various missing rates

Table 6 Forecasting average performance on the Electricity test set under all missing rates

Method	Extrap		
	MSE	RMSE	MAE
Latent ODE	0.2935	0.5353	0.3815
GRU-VAE	0.2038	0.4482	0.3149
ODERNN-VAE	0.2136	0.4620	0.3183
TN-ODE (ours)	0.1704	0.4128	0.2859

decline primarily results from the RNN-based models' inability to capture complete contextual information as the missing rates increase, making it challenging to predict the next data point accurately. However, BRITS can learn the sequence from both directions, effectively alleviating the problem of data missing. The model training data decreases with the increase of the missing rate, and there is a higher likelihood of underfitting or overfitting.

Moreover, the interpolation ability of ODE-based models remains stable, primarily because they can leverage existing data points to interpolate missing ones, and the interpolation error can be optimized by adaptively adjusting the step size and

accuracy through the ODE solver. Among these models, the TN-ODE model has the lowest MSE values, which can be attributed to the time-aware encoder's superior ability to learn the posterior distribution from sparse data. The AJ-RNN model also has a stable imputation ability, the performances of which is much poorer than that of the proposed model. As the RMSE, and MAE metrics shown in Fig. 4b, c, these metrics consistent with the trend of the MSE results.

For the average performance across different missing rates (presented in Table 5), our TN-ODE model is only inferior to the BRITS model, as the BRITS model performs well when the missing rate is low, while our model performs even better when missing rates are high.

The extrapolation task is still verified on the datasets with missing rates varying from 0 to 70%. Figure 5 describes the extrapolation performance of ODE-based methods on electricity dataset with missing rates ranging from 0 to 70% in terms of MSE, RMSE, and MAE. The TN-ODE model consistently outperforms the other encoder methods including the RNN, GRU, ODE-RNN encoders in most cases. Table 6 shows the average forecasting results on electricity test set of all missing rates, demonstrating that the TN-ODE model achieves the best result of extrapolation. The time-aware LSTM encoder can effectively learn the posterior distribution.

5.4 Performance comparison for the downstream application

The classification task is trained on the human activity dataset using the TN-ODE model and baseline models. The data are collected from five individuals by four inertial measurement sensors attached to their belt, chest and ankles (12 features in total). The data is processed as the paper [13] does, which has 6554 sequences of 211 time points. The objective of this task is to classify the current activity of each individual into one of seven types of activities (walking, sitting, etc.). For performance evaluation of the proposed model, a series of comparative experiments are carried out. Figure 6 presents the detailed test classification accuracy using the

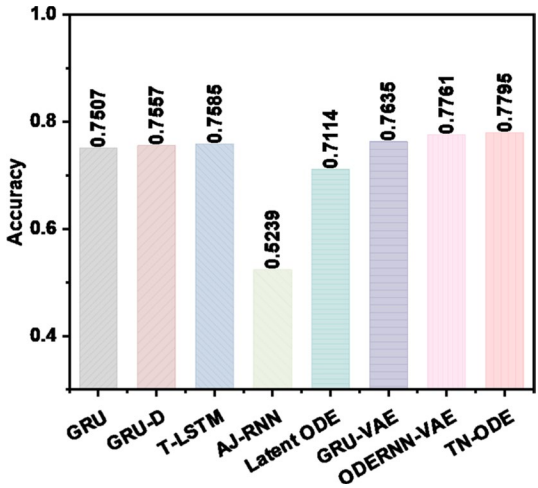

Fig. 6 The detailed test classification accuracy of UCI real incomplete Human Activity Dataset

Table 7 Imputation and prediction results of ablated models on the Sine Wave Simulation

Model	Interp			Extrap		
	MSE	RMSE	MAE	MSE	RMSE	MAE
T-LSTM	0.0327	0.1809	0.1224	–	–	–
LSTM-VAE	0.0214	0.1462	0.1058	0.0043	0.0659	0.0465
TN-ODE (ours)	0.0106	0.1027	0.0729	0.0021	0.0457	0.0317

Table 8 Imputation and prediction results of ablated models on the PhysioNet Dataset

Model	Interp			Extrap		
	MSE	RMSE	MAE	MSE	RMSE	MAE
T-LSTM	0.0104	0.0627	0.0576	–	–	–
LSTM-VAE	0.0071	0.0496	0.0450	0.0046	0.0339	0.0312
TN-ODE (ours)	0.0059	0.0459	0.0412	0.0041	0.0308	0.0288

proposed model and corresponding comparative algorithms on UCI real incomplete human activity dataset. The TN-ODE classifier has higher accuracy than classifiers of RNN variants, AJ-RNN and ODE-based classifiers with other encoders on this task. The TN-ODE model can effectively generate continuous-time latent states according to the posterior that inferred from irregularly-sampled time series dataset. The time-aware LSTM encoder can decrease error accumulation and avoid the issue of vanishing gradient. While the significance of the classification test on a single dataset is limited, it still shows great potential of generalization ability of the TN-ODE model.

5.5 Ablation study

This paper also evaluates the contributions of the time-aware encoder and the Neural ODE model by ablation study on the Sine Wave Simulation, PhysioNet Dataset, Electricity Dataset and Human Activity Dataset. Tables 5, 6, 7, 8 and 11 present the results of the ablated proposed models. This paper compares the following ablated models:

Ours: the proposed model

T-LSTM: the model without Neural ODEs

LSTM-VAE: the model employs an LSTM as an encoder without the function of time perception

Table 7 shows the imputation and prediction performances of ablated models on the Sine Wave Simulation (measured by MSE, RMSE and MAE). The first row only shows the imputation results of T-LSTM due to the T-LSTM model does not support long-time prediction. The second row presents the performances of LSTM-VAE, and the last row is the proposed model. As can be seen, the proposed model outperforms ablated models. The Neural ODE model can help to forecast the future system dynastics. And the time-aware encoder can effectively learn the posterior distribution from irregularly-sampled time series.

Table 9 Imputation results of ablated models on the Electricity Dataset with different missing rates

Method	Index	Missing rate (%)							
		0	10	20	30	40	50	60	70
T-LSTM	MSE	0.0835	0.1936	0.2955	0.3946	0.4886	0.5729	0.6571	0.7431
	RMSE	0.2890	0.4399	0.5436	0.6382	0.6990	0.7569	0.8106	0.8620
	MAE	0.2043	0.2791	0.3471	0.4129	0.4766	0.5359	0.5947	0.6540
LSTM-VAE	MSE	0.1639	0.1693	0.1918	0.1692	0.1900	0.1951	0.1758	0.2227
	RMSE	0.4039	0.4115	0.4380	0.4113	0.4359	0.4418	0.4193	0.4719
	MAE	0.2806	0.2870	0.3051	0.2871	0.3063	0.3073	0.2933	0.3232
TN-ODE	MSE	0.1607	0.1692	0.1711	0.1632	0.1651	0.1714	0.1728	0.1821
	RMSE	0.4009	0.4113	0.4137	0.4040	0.4046	0.4140	0.4157	0.4267
	MAE	0.2779	0.2869	0.2890	0.2815	0.2836	0.2874	0.2887	0.2972

Table 10 Forecasting prediction results of ablated models on the Electricity Dataset with a variety of missing rates

Method	Index	Missing rate (%)							
		0	10	20	30	40	50	60	70
LSTM-VAE	MSE	0.1910	0.1782	0.1779	0.1786	0.1712	0.1729	0.1763	0.1796
	RMSE	0.4370	0.4221	0.4218	0.4227	0.4158	0.4116	0.4187	0.4238
	MAE	0.3001	0.2910	0.2894	0.2934	0.2871	0.2844	0.2891	0.2920
TN-ODE	MSE	0.1753	0.1645	0.1699	0.1670	0.1711	0.1683	0.1691	0.1779
	RMSE	0.4187	0.4055	0.4122	0.4086	0.4138	0.4102	0.4112	0.4218
	MAE	0.2918	0.2813	0.2866	0.2818	0.2868	0.2842	0.2841	0.2902

Table 8 shows the ablation experiments of TN-ODE on the PhysioNet Dataset. Experimental results illustrate that TN-ODE exhibits the best accuracy in both interpolation and prediction tasks. The experimental results are closely related to each component of the proposed model. The time-aware encoder helps to learn an accuracy posterior distribution. The Neural ODE module can learn the system dynamics and forecast continuous-time latent states.

Tables 9 and 10 display the imputation and forecasting results on the electricity dataset,respectively. The experimental results are consistent with the sine wave simulation. The proposed model outperforms ablated models, indicating that the time-aware encoder has an important role to effectively perceive the time interval and the Neural ODE model is capable of learning system dynamics.

Table 11 exhibits the classification results of ablated models on human activity dataset. Results demonstrate that TN-ODE achieves the best performance among ablated models. As can be observed, the function of time perception and Neural ODEs are both helpful. The classification accuracy decreases 2.51% without the function of time perception and 2.1% without Neural ODEs.

Table 11 Classification results of ablated models on Human Activity Dataset

Model	Accuracy
T-LSTM	0.7585
LSTM-VAE	0.7544
TN-ODE (ours)	0.7795

5.6 Runtime analysis

Table 12 describes the time consumption of the TN-ODE model and comparison algorithms for the interpolation task. The results show that ODE-based models take more time than RNN-based models. This is because RNN-based models only predict one-step-ahead, whereas ODE-based models with a VAE structure need to reconstruct the whole trajectories given the vector summary z0. Among ODE-based methods, the RNN encoder has the shortest training time, and our time-aware encoder requires more time than the RNN encoder and less time than the ODE-RNN encoder. These findings indicate the additional use of ODEs inevitable increases training time.

6 Conclusion

In this article, a novel TN-ODE is proposed for modeling incomplete time series. TN-ODE not only provides imputation for missing values as well as multi-step data prediction via an ODE network, but also supports classification tasks. In TN-ODE, a time-aware LSTM is adopted to learn temporal and sequential features of incomplete time series without the need for data preprocessing. The ODE network is responsible for generating continuous-time latent dynamics by learning the relationship between unknown systems and their derivatives. Results demonstrate the TN-ODE model outperforms baseline algorithms, which is attributed to the time-aware LSTM being able to learn an approximate posterior and providing more adaptive input representations for the ODE network. Future work will explore modifications of the encoder in this continuous model by using bidirectional recurrent networks.

Table 12 Time consumption of eight approaches on the interpolation task

Method dataset	Sine wave simulation	PhysioNet	Electricity
GRU	0.0184	1.6902	0.02913
GRU-D	0.0453	21.5342	6.2668
T-LSTM	0.0431	4.7324	0.2783
BRITS	0.0492	0.2545	0.0527
AJ-RNN	0.0208	2.1539	0.0586
Latent ODE	0.0420	3.6128	0.2479
ODERNN-VAE	0.1266	7.7247	0.3754
TN-ODE (ours)	0.0727	6.2943	0.3357

Additionally, attention mechanisms will be used to learn temporal information of time series. Lastly, combination Neural ODEs with generative adversarial networks to model incomplete time series remains to be investigated in future work.

Acknowledgements This work was supported by the Major Projects of Technical Innovation of Hubei Province under Grant 2018AAA046.

Author contributions Zhuoqing Chang: Conceptualization, Methodology, Software and Writing-Original Draft. Shubo Liu: Resources, Project administration and Funding acquisition. Run Qiu: Software and Validation. Song Song: Writing-Review & Editing. Zhaohui Cai: Supervision and Validation. Guoqing Tu: Formal analysis and Investigation.

Availability of data and materials The datasets generated during and/or analysed during the current study are available in the ElectricityLoadDiagrams20112014 and Localization+Data+for+Person+Activity repository, https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014 and https://archive.ics.uci.edu/ml/datasets/Localization+Data+for+Person+Activity/.

Code availability The code required to reproduce these findings cannot be shared at this time as the code also forms part of an ongoing study.

Declarations

Conflict of interest The authors declare that the publication of this paper has no conflicts of interest.

Ethics approval This paper is not submitted to any other journals. On behalf of all authors, I have been very impressed by the overall quality of the work published by Nature.

Consent to participate All authors have approved the manuscript for submission and without any potential competing interests.

Consent for publication This manuscript describes original work and is not under consideration by any other journal. All authors approved the manuscript and this submission.

References

1. Chang Z, Liu S, Xiong X, Cai Z, Tu G (2021) A survey of recent advances in edge-computing-powered artificial intelligence of things. IEEE Internet Things J 8(18):13849–13875
2. Pan Z, Wang Y, Wang K, Chen H, Yang C, Gui W (2023) Imputation of missing values in time series using an adaptive-learned median-filled deep autoencoder. IEEE Trans Cybern 53(2):695–706. https://doi.org/10.1109/TCYB.2022.3167995
3. Wang T, Ke H, Jolfaei A, Wen S, Haghighi MS, Huang S (2022) Missing value filling based on the collaboration of cloud and edge in artificial intelligence of things. IEEE Trans Ind Inf 18(8):5394–5402. https://doi.org/10.1109/TII.2021.3126110
4. Folino G, Pisani FS (2016) Evolving meta-ensemble of classifiers for handling incomplete and unbalanced datasets in the cyber security domain. Appl Soft Comput 47:179–190
5. Batista GE, Monard MC (2003) An analysis of four missing data treatment methods for supervised learning. Appl Artif Intell 17(5–6):519–533
6. Box GE, Jenkins GM, Reinsel GC, Ljung GM (2015) Time series analysis: forecasting and control. Wiley, Hoboken
7. Hastie T, Mazumder R, Lee JD, Zadeh R (2015) Matrix completion and low-rank SVD via fast alternating least squares. J Mach Learn Res 16(1):3367–3402
8. Bengio Y, Gingras F (1995) Recurrent neural networks for missing or asynchronous data. Adv Neural Inf Process Syst 8

9. Lipton ZC, Kale D, Wetzel R (2016) Directly modeling missing data in sequences with rnns: improved classification of clinical time series. In: Machine Learning for Healthcare Conference, pp 253–270
10. Che Z, Purushotham S, Cho K, Sontag D, Liu Y (2018) Recurrent neural networks for multivariate time series with missing values. Sci Rep 8(1):1–12
11. Choi E, Bahadori MT, Schuetz A, Stewart WF, Sun J (2016) Doctor ai: Predicting clinical events via recurrent neural networks. In: Machine Learning for Healthcare Conference, pp 301–318
12. Cao W, Wang D, Li J, Zhou H, Li L, Li Y (2018) Brits: bidirectional recurrent imputation for time series. Adv Neural Inf Process Syst 31
13. Chen RTQ, Rubanova Y, Bettencourt J, Duvenaud DK (2018) Neural ordinary differential equations. In: Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N, Garnett R (eds) Advances in Neural Information Processing Systems, vol 31. Montréal
14. Rubanova Y, Chen RTQ, Duvenaud DK (2019) Latent ordinary differential equations for irregularly-sampled time series. In: Wallach H, Larochelle H, Beygelzimer A, d' Alché-Buc F, Fox E, Garnett R (eds) Advances in Neural Information Processing Systems, vol 32. Vancouver, BC
15. Amiri M, Jensen R (2016) Missing data imputation using fuzzy-rough methods. Neurocomputing 205:152–164
16. Li SC, Jiang B, Marlin BM (2019) Misgan: learning from incomplete data with generative adversarial networks. In: 7th International Rence on Learning Representations, ICLR 2019, New Orleans, LA, USA,Conference May 6–9, 2019, New Orleans, LA
17. Yoon J, Jordon J, van der Schaar M (2018) GAIN: Missing data imputation using generative adversarial nets. In: Dy J, Krause A (eds) Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol 80, pp 5689–5698. Stockholm
18. Luo Y, Zhang Y, Cai X, Yuan X (2019) E^2gan: end-to-end generative adversarial network for multivariate time series imputation. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, pp 3094–3100. International Joint Conferences on Artificial Intelligence Organization, Macao, SAR
19. Wu Z, Ma C, Shi X, Wu L, Dong Y, Stojmenovic M (2022) Imputing missing indoor air quality data with inverse mapping generative adversarial network. Build Environ 215:108896
20. Ma Q, Li S, Cottrell GW (2022) Adversarial joint-learning recurrent neural network for incomplete time series classification. IEEE Trans Pattern Anal Mach Intell 44(4):1765–1776
21. Lipton ZC, Kale D, Wetzel R (2016) Directly modeling missing data in sequences with rnns: Improved classification of clinical time series. In: Doshi-Velez F, Fackler J, Kale D, Wallace B, Wiens J (eds) Proceedings of the 1st Machine Learning for Healthcare Conference. Proceedings of Machine Learning Research, vol 56, pp 253–270. Northeastern University, Boston, MA, USA. https://proceedings.mlr.press/v56/Lipton16.html
22. Choi E, Bahadori MT, Schuetz A, Stewart WF, Sun J (2016) Doctor ai: predicting clinical events via recurrent neural networks. In: Doshi-Velez F, Fackler J, Kale D, Wallace B, Wiens J (eds) Proceedings of the 1st Machine Learning for Healthcare Conference. Proceedings of Machine Learning Research, vol 56, pp 301–318. Northeastern University, Boston, MA, USA
23. Baytas IM, Xiao C, Zhang X, Wang F, Jain AK, Zhou J (2017) Patient subtyping via time-aware lstm networks. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '17. Association for Computing Machinery, New York, NY, USA, pp 65–74
24. Cao W, Wang D, Li J, Zhou H, Li L, Li Y (2018) Brits: bidirectional recurrent imputation for time series. In: Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N, Garnett R (eds) Advances in Neural Information Processing Systems, vol 31. Curran Associates Inc, Montréal
25. Shukla SN, Marlin BM (2021) Multi-time attention networks for irregularly sampled time series. arXiv preprint arXiv:2101.10318
26. Sun C, Hong S, Song M, Zhou Y, Sun Y, Cai D, Li H (2021) Te-esn: time encoding echo state network for prediction based on irregularly sampled time series data. arXiv preprint arXiv:2105.00412
27. Dupont E, Doucet A, Teh YW (2019) Augmented neural odes. In: Wallach H, Larochelle H, Beygelzimer A, d' Alché-Buc F, Fox E, Garnett R (eds) Advances in Neural Information Processing Systems, vol 32. Curran Associates, Inc., Vancouver, BC
28. Schirmer M, Eltayeb M, Lessmann S, Rudolph M (2022) Modeling irregular time series with continuous recurrent units. In: International Conference on Machine Learning, pp 19388–19405
29. Chen Y, Li Z, Yang C, Wang X, Long G, Xu G (2022) Adaptive graph recurrent network for multivariate time series imputation. In: International Conference on Neural Information Processing

30. Guo J, Zhang P, Li C, Xie X, Zhang Y, Kim S (2022) Evolutionary preference learning via graph nested gru ode for session-based recommendation. In: Proceedings of the 31st ACM International Conference on Information and Knowledge Management, pp 624–634
31. Zhang X, Zeman M, Tsiligkaridis T, Zitnik M (2021) Graph-guided network for irregularly sampled multivariate time series. arXiv preprint arXiv:2110.05357
32. Huang Z, Sun Y, Wang W (2021) Coupled graph ode for learning interacting system dynamics. In: The 27th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)
33. Huang Z, Sun Y, Wang W (2020) Learning continuous system dynamics from irregularly-sampled partial observations. Adv Neural Inf Process Syst 33:16177–16187
34. Kingma DP, Welling M (2013) Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114
35. Contreras-Reyes JE (2014) Asymptotic form of the Kullback–Leibler divergence for multivariate asymmetric heavy-tailed distributions. Physica A 395:200–208
36. Pontryagin LS (ed) (1987) Mathematical Theory of Optimal Processes. CRC Press, Switzerland
37. Zhang Y (2019) Attain: attention-based time-aware lstm networks for disease progression modeling. In: In Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI-2019), pp 4369–4375, Macao, China
38. Silva I, Moody G, Scott DJ, Celi LA, Mark RG (2012) Predicting in-hospital mortality of ICU patients: the physionet/computing in cardiology challenge 2012. In: 2012 Computing in Cardiology, pp 245–248. IEEE
39. Artur T (2015) ElectricityLoadDiagrams20112014 Data Set. https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014. Accessed 2015
40. Mitja L, Bostjan K, Rok P, Jana K, Vedrana V (2010) Localization Data for Person Activity Data Set. https://archive.ics.uci.edu/ml/datasets/Localization+Data+for+Person+Activity/. Accessed 2010
41. Cho K, van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, Doha, Qatar, pp 1724–1734

Authors and Affiliations

Zhuoqing Chang[1] · Shubo Liu[1] · Run Qiu[1] · Song Song[1] · Zhaohui Cai[1] · Guoqing Tu[2]

Zhuoqing Chang
changzhuoqing@whu.edu.cn

Run Qiu
qiurun@whu.edu.cn

Song Song
song.s@whu.edu.cn

Zhaohui Cai
zhcai@whu.edu.cn

Guoqing Tu
tugq2000@163.com

[1] School of Computer Science, Wuhan University, 299# Bayi Rd, Wuchang District, Wuhan 430072, Hubei, China

[2] School of Cyber Science and Engineering, Wuhan University, 299# Bayi Rd, Wuchang District, Wuhan 430072, Hubei, China