Deep Efficient Continuous Manifold Learning for Time Series Modeling

Seungwoo Jeong, Student Member, IEEE, Wonjun Ko, Ahmad Wisnu Mulyadi, and Heung-II Suk, Senior Member, IEEE

Abstract—Modeling non-euclidean data is drawing extensive attention along with the unprecedented successes of deep neural networks in diverse fields. Particularly, a symmetric positive definite matrix is being actively studied in computer vision, signal processing, and medical image analysis, due to its ability to learn beneficial statistical representations. However, owing to its rigid constraints, it remains challenging to optimization problems and inefficient computational costs, especially, when incorporating it with a deep learning framework. In this paper, we propose a framework to exploit a diffeomorphism mapping between Riemannian manifolds and a Cholesky space, by which it becomes feasible not only to efficiently solve optimization problems but also to greatly reduce computation costs. Further, for dynamic modeling of time-series data, we devise a continuous manifold learning method by systematically integrating a manifold ordinary differential equation and a gated recurrent neural network. It is worth noting that due to the nice parameterization of matrices in a Cholesky space, training our proposed network equipped with Riemannian geometric metrics is straightforward. We demonstrate through experiments over regular and irregular time-series datasets that our proposed model can be efficiently and reliably trained and outperforms existing manifold methods and state-of-the-art methods in various time-series tasks.

Index Terms—Cholesky space, deep learning, manifold learning, manifold ordinary differential equation, multivariate time series modeling, symmetric positive definite matrix.

I. INTRODUCTION

Time series data is a set of sequential observation values over time that is ubiquitous in the real-world such as video, natural language, audio, bio-signals, etc. For example, in computer vision, a video is a set of consecutive images that await to be explored in diverse tasks such as action recognition [1], object tracking [2], etc. In natural language processing, a sequence of words is used for tasks of document classification [3], sentiment analysis [3], and language translation [4]. Furthermore, electroencephalogram (EEG) and electrocardiogram (ECG) data are bio-signals, frequently used for diagnostic-related tasks, such as sleep staging [5] and seizure detection [6], respectively. Recently, with the development of devices, it has become possible to densely record more complex observations, and accordingly, more accurate and effective modeling of time-series data is essential.

While there has been a number of literature on algorithms and pattern analysis methods, deep learning has recently been most exhaustively employed as the de facto standard for time-series modeling. These approaches use convolutional neural networks (CNNs) or recurrent neural networks (RNNs) to capture spatial or temporal features of time-series data [9], [10], [20], [21], [22]. Recently, Transformer-based methods [23], [24] have been used to handle the lack of long-range dependency, which is one of the problems of RNN. However, these neural network-based methods suffer from (i) an inadequacy in handling non-euclidean characteristics in data such as processing data in manifold space and (ii) insufficient data for processing complex and high-dimensional data. In this context, utilizing manifold space for modeling in deep learning for time series data offers several notable advantages compared to traditional euclidean approaches. By effectively capturing the intrinsic non-linearity inherent in time series data and employing a mapping procedure onto lower-dimensional manifold, deep learning models become capable of adeptly capturing intricate temporal patterns and dependencies that would otherwise go unnoticed in euclidean representations [25], [26], [27]. Consequently, this facilitates the attainment of more precise and expressive data representations. Furthermore, the adoption of manifold space learning engenders heightened robustness of models to the influences of noise and outliers [26], an augmented capacity for generalization through an emphasis on crucial variations [27], hence providing a more condensed and abstract perspective of temporal patterns. These advantages collectively contribute to advancing more dependable, efficient, and interpretable models for time series analysis.

Simultaneously, researchers have actively explored manifold space as a computational method for efficiently handling the geometric features of data. Such approaches are often used for the analysis of various representations, including but not limited to shapes [28], graphs and trees [29], [30], symmetric positive
require a technique that considers the characteristics of the manifold space. Of these data types, the SPD matrix has received considerable attention in a variety of application areas due to its ability to learn suitable statistical representations [34], [35], [36]. In computer vision, SPD matrices have achieved great success as representations for classification tasks such as skeletal-based human behavioral recognition [37] and emotion recognition [34]. In signal processing, it has also been used as an expression method for multivariate time-series (MTS) data [35]. In neuroimage analysis, a diffusion tensor image, which indicates structural connections over brain regions, has been treated as manifold-valued data [36]. Nevertheless, the direct use of SPD matrices remains a challenge because of its rigid constraints. SPD matrices in the euclidean metric cause unfavorable results, such as the swelling effect [38] and a finite distance of symmetric matrices with non-positive eigenvalues [39], [40]. Thus, it is unreasonable for conventional deep-learning methods to model SPD matrices by ignoring their geometric characteristics and computational constraints. Additionally, backpropagation in the optimization process makes it difficult for operand components within deep neural networks to maintain positive definiteness. To address these problems, several methods have been proposed to train the model while maintaining the structure of the SPD matrix. For instance, SPDNet [34] handles this issue through an orthogonal weight matrix on the compact Stiefel manifolds. In [41], the optimizer structure was designed through a meta-learning-based method. Furthermore, existing deep learning methodologies, e.g., CNN-based models [1], [36] or RNN-based models [17], were proposed to deal with the manifold-valued sequential data, suggesting a recursive method for the weighted Fréchet mean to avoid the problems of high computational cost and instability in optimization. However, numerical errors and excessive costs remain challenges in the computation process.

Generally, modeling time-series data is based on the premise of discrete observations and emission intervals. However, data in the real world are mainly irregularly-sampled data possibly due to unexpected errors in recording or device. Fig. 1 presents an example of time-series data with one missing value at a time point $t_3$ on a discrete-time interval. When modeling and handling the missing value using an imputation technique, e.g., bi-linear interpolation, in a euclidean metric (green-colored arrows), it causes a large error compared to the one estimated in a manifold metric. Even when given two data points with the same interval, its latent representations in the manifold take the form of a curve (rather than a linear form) due to the geometric characteristics. Notably, it is beneficial to model in a continuous-time space so that the geometric features inherent in the data are better represented, thus improving target downstream tasks. Therefore, it is of paramount importance to develop a continuous manifold learning method for time-series modeling.

A recent provocative approach to deal with irregularly sampled data is a continuous-time model through ordinary differential equations (ODE) with a time variable [42]. Neural ODE [42] and its variants [12], [13] have demonstrated their capability in signal representation by capturing dynamics over the internal hidden states. Subsequent studies extend the neural ODE to the manifold space [18], [19]. For example, Manifold ODE [18] has been proposed to learn the dynamics of the states in the Riemannian manifold, aiming to estimate complicated distributions in the framework of normalizing flow [43]. Although various neural ODEs for manifold space have been proposed, their application for time-series data is still limited and, even if applicable, computationally expensive and inefficient.

In this work, we propose a novel method of continuous manifold learning on sequential manifold-valued data for time-series modeling. Based on rigorous mathematical justifications, we propose to exploit a diffeomorphism mapping between Riemannian manifolds and a Cholesky space [40], by which it becomes feasible to efficiently solve optimization problems and to greatly reduce computational costs. We devise an algorithm to solve an optimization problem under the SPD constraint and achieve an effective computation cost by re-defining operations within a deep neural network framework. Our proposed method represents manifold-valued sequential data via a practical computation based on well-defined mathematical formulations. Particularly, we introduce a recurrent neural model on the Cholesky space, making it possible to greatly enhance computational efficiency and train network parameters with conventional backpropagation. Moreover, the proposed method successfully learns real-world continuous data modeling for irregularity or sparsity and adjusts the discrepancy in geodesics with the use of a manifold ODE for dynamics representation utilizing an RNN-based network. To prove the validity and robustness of our proposed method, we conducted exhaustive experiments and achieved promising performance over various public datasets.

In summary, there are three major contributions of this work:

- To handle the hard constraints of an SPD matrix in time-series modeling, we devise a novel computational mechanism that exploits a diffeomorphism mapping from SPD space to Cholesky space and applies it to a deep recurrent network. Our proposed method greatly reduces computational costs and numerical errors, thus enabling robust and stable learning.
- We also develop a manifold ODE representing temporal dynamics inherent in time-series data, allowing it to learn trajectory with geometric characteristics of sequential manifold-valued data.
- We exhaustively demonstrate the effectiveness of the proposed method through experiments over various public datasets, achieving state-of-the-art performance.
We organized the rest of this work as follows. Section II briefly reviews related work. In Section III, we describe essential preliminary concepts to pave the way for understanding the proposed method. Then, Section IV introduces the proposed method for continuous manifold learning with the diffeomorphism estimation. Section V describes experimental settings in detail for reproducibility and results are compared with existing methods. Finally, we conclude the study and recommend future research directions in Section VI.

II. RELATED WORK

In this section, we introduce existing research on discrete and continuous time-series modeling. Table I compares recent methods with our proposed in terms of computing space, continuous, and time regularity.

A. Discrete Time-Series Modeling

Most general models employed as baselines in time-series data are distance function-based methods [44]. Traditionally, nearest neighbor (NN) classifiers with distance functions [45] have been widely used, and the method used with Dynamic Time Warping (DTW) in particular has shown good performance [46]. In WEASEL-MUSE [8], a bag of Symbolic Fourier Approximation (SFA) is used for MTS classification. However, these classical models have limited scope and may face challenges when dealing with complex patterns, as well as limitations in terms of computational cost, sensitivity to outliers, and potential data loss.

Recently, deep learning-based methods have shown promising performance. In MLSTM-FCN [21], a model combining long short-term memory (LSTM) and CNN is proposed for MTS classification tasks. Specifically, performance is improved by adding squeeze-and-excitation blocks [47]. In [22], an unsupervised method is proposed to solve the variable lengths and sparse labeling problems of time-series data. The authors process the variable length through a dilated convolutional [48] encoder and build a triplet loss [49] with negative samples. Meanwhile, for high-dimensional multivariate data and limited training label problems, TapNet [9] proposes an attentional prototype network by learning low-dimensional features through random group permutation. In ShapeNet [10], the author proposes an MDC-CNN with a triplet loss that trains time-series subsequences of various lengths in a unified space. Meanwhile, transformer-based methods [23, 24] have been mainly used for forecasting time-series data, especially due to their ability to model long-range dependencies. Representation learning for time series data has garnered significant attention in recent years. Of various techniques, contrastive learning-based methods have emerged as prominent approaches, offering universal representation learning for various time series data tasks [11, 24]. Zerveas et al. [24] propose an unsupervised representation learning method based on transformers, evaluating its effectiveness in multivariate time series regression and classification. In [11], TS2Vec is introduced as a technique for learning universal representations at different semantic levels. Although these models use deep learning-based methods to learn spatio-spectral-temporal features from time-series data, they are limited in the assumption that the data have constant intervals and thus are most suitable for regular time-series data. In contrast, our proposed method is a neural ODE-based method that enables continuous time-series modeling.

B. Continuous Modeling

Meanwhile, researchers have focused on modeling dynamic patterns at continuous time points. Foremost, [42] proposes the idea of parameterizing derivatives of hidden states for continuous modeling. Inspired by neural ODEs, [12] propose latent ODE and ODE-RNN to deal with irregularly-sampled data. In [13], GRU-ODE-Bayes, combining GRU-ODE and GRU-Bayes, is proposed to address sporadic observations that occur in multidimensional time-series data. Theoretically similar to [13], in Neural CDE [16], an extended model of Neural ODE using a controlled differential equation is proposed to address the limitations of an ordinary differential equation. Meanwhile, in ODE2VAE, by decomposing the latent space into position and velocity components, time-sequential modeling for sequential data is proposed [15]. In [50], for the relevance or importance of individual samples, a continuous-time attention method is proposed by applying the attention mechanism to the neural ODE.

Recent studies on the use of neural ODEs have begun to model structured [51] or manifold-valued data [18]. To capture
the essence of continuous node representations, [51] introduces a graph-based model. Within the research community, neural ODEs have been expanded from their original domain of euclidean space to encompass a manifold space, particularly in normalizing flow tasks [18], [19]. Even with the potential demonstrated by these advancements, a substantial gap exists in research regarding applying neural ODEs to high-dimensional time-series data. In contrast, our model takes a different approach by employing neural ODEs to specifically address continuous time-series modeling, focusing on the trajectories of hidden states within a manifold space while incorporating structural features.

### C. Manifold Learning

In order to better represent the geometric characteristics inherent in time-series data, recent studies focus on modeling with manifold-valued data, especially SPD matrices, in deep learning frameworks. In [34], the author proposes an SPDNet that consists of fully connected convolution-like layers, rectified linear units-like layers, and eigenvalue logarithm layers to train the deep model while maintaining the data structure on the manifold space. Subsequently, [52] redefines the batch normalization on SPDNet using Riemannian operations to improve the classification performance and robustness of the model. In [35], the author proposes a metric learning-based SPD model that divides the classification problem into sub-problems and builds a single model to solve each sub-problem. Succinctly, SPDNet models are presented based on mapping with sub-manifold, but the repeated eigenvalue decomposition in the process inevitably generates tremendous computation costs.

Meanwhile, [17] proposes a deep model based on statistical machine learning when data are ordered, longitudinal, or temporal in a Riemannian manifold. Additionally, to resolve the slow speed issue of RNN-based models, the dilated and causal CNNs were defined in the manifold space [1], [36]. In [53], a flow-based generative model for manifold-valued data is introduced through three types of invertible layers in a generative regime. Training a deep model while maintaining the SPD constraint is challenging due to huge computational costs and unstable training.

Our approach differs from these previous SPD manifold methods, which directly process the SPD matrix. We accomplished this through mapping into the Cholesky space, which is easily to compute. This way, our proposed method relaxes the constraints enough to allow the optimization to be processed with the training algorithms.

### III. PRELIMINARIES

In this section, we concisely provide useful definitions and operations for Riemannian geometry. The notations and operations in Riemannian and Cholesky spaces are summarized in Table II. Note that for the distance in Riemannian space, we denote the affine-invariant Riemannian metric.

#### A. Riemannian Manifold and Metric on Cholesky Space [40]

A Riemannian manifold \((M, g)\) is a real, smooth manifold \(M\) equipped with a positive-definite inner product \(g_X\) on the tangent space \(T_X M\) at each point \(X \in M\). The \(g_X\) is called a Riemannian metric, making it possible to define several geometric notions on a Riemannian manifold, such as the angle and curve length.

The space \(S_d^+\) of \(d \times d\) SPD matrices is a smooth submanifold of the space \(S_d\) of symmetric matrices, whose tangent space at a given SPD matrix is identified with \(S_d\). Cholesky decomposition represents a matrix \(S \in S_d^+\) as a product of a lower triangular matrix \(L\) and its transpose, i.e., \(S = LL^T\). The smooth submanifold \(\mathcal{L}_+\) of the space of lower triangular matrices \(\mathcal{L}\) with diagonal elements all positive defines a Cholesky space. Note that the tangent space of \(\mathcal{L}_+\) at a given matrix \(L \in \mathcal{L}_+\) is identified with the linear space \(\mathcal{L}\). Let the strict lower triangular space \(\mathcal{L} = \{X | X \in \mathcal{L}\} \in \mathbb{R}^{d(d-1)/2}\) with the Frobenius inner product \(\langle X, Y \rangle_F = \sum_{i=1}^{d} X_{ij}Y_{ij}, \forall X, Y \in \mathcal{L}\), while denoting the diagonal part \(\mathcal{D}(L) = \{D(X)|X \in \mathcal{L}\}\) with a different inner product \((\mathcal{D}(L)^{-1}D(X), \mathcal{D}(L)^{-1}D(Y))_F\).

Then, we define a metric \(\tilde{g}\) for the tangent space \(T_L \mathcal{L}_+\) as follows,

\[
\tilde{g}_L(X, Y) = \langle X, Y \rangle_F + \langle \mathcal{D}(L)^{-1}D(X), \mathcal{D}(L)^{-1}D(Y) \rangle_F
\]

\[
= \sum_{i=1}^{d} \sum_{j=1}^{i} X_{ij}Y_{ij} + \sum_{j=1}^{d} X_{jj}Y_{jj}L_{jj}^{-2}.
\]

**Proposition 1:** [40] Cholesky map \(\mathcal{L}\) is a diffeomorphism between smooth manifolds \(\mathcal{L}_+\) and \(S_d^+\).

The full proof of Proposition 1 can be found in [40]. Essentially, the Cholesky map easily represents deep learning processes in the SPD manifold space.

### TABLE II

**Summary of Operations in the Riemannian Space and the Cholesky Space**

| Manifold | Riemannian space | Cholesky space |
|----------|-----------------|---------------|
| \(S_d^+\) | \(\{P \in \mathbb{R}^{d \times d} : P = P^\top, X^\top PX > 0, \forall X \in \mathbb{R}^d \setminus \{0_d\}\}\) | \(\{X \in \mathbb{R}^{d \times d} : \mathcal{D}(X) > 0, \{X_{ij}\}_{i,j > 1} = 0\}\) |
| Metric | \(T_X M \times T_X M\) | \(\sum_{i,j>1} X_{ij}Y_{ij} + \sum_{i=1}^{m} X_{ij}Y_{ij}L_{jj}^{-2}\) |
| Distance | \(\frac{1}{2}||\log(P_{XX}^{-\frac{1}{2}}P_{YY}^{-\frac{1}{2}})||_F\) | \(\{||L|| - ||K|| \}^2 + ||\log \mathcal{D}(L) - \log \mathcal{D}(K)\}^2 \frac{1}{2}\) |
| Exponential map | \(P^\frac{1}{2} \exp(P^\frac{1}{2} Q^{-\frac{1}{2}}) P^\frac{1}{2}\) | \(|X| + |K| + \mathcal{D}(K) \exp(\mathcal{D}(K) \mathcal{D}(X)^{-1})\) |
| Logarithmic map | \(Q^\frac{1}{2} \log(Q^\frac{1}{2} P^{-\frac{1}{2}}) Q^\frac{1}{2}\) | \(|X| - |K| + \mathcal{D}(K) \log(\mathcal{D}(K)^{-1}\mathcal{D}(X))\) |

The notations used in the operations are as follows; \(d\): the dimension of a matrix, \(P\): a point in Riemannian space, \(Q\): a point in tangent space of Riemannian space, \(X, Y\): points in Cholesky space, \(K\): a point in tangent space of Cholesky space. Note that for the distance in Riemannian space, we denote the affine-invariant Riemannian metric.
B. Exponential and Logarithmic Maps

For any \( P \in S^+_d \) and \( Q \in S_d \), the exponential map \( \text{Exp}_P : S_d \rightarrow S^+_d \) and the logarithmic map \( \text{Log}_Q : S^+_d \rightarrow S_d \) are defined as

\[
\text{Exp}_P(Q) = P^{1/2} \exp (P^{1/2} Q P^{-1/2}) P^{1/2} \in S^+_d, \quad (2)
\]

\[
\text{Log}_Q(P) = Q^{1/2} \log (Q^{1/2} P Q^{-1/2}) Q^{1/2} \in S_d. \quad (3)
\]

Generally, it is difficult to compute Riemannian exponential and logarithmic maps because they require evaluating a series of infinite number of terms [39]. However, by mapping to Cholesky space, we obtain easy-to-compute expressions for Riemannian exponential and logarithmic maps. Let \( X \in \mathcal{L}_+ \) and \( K \in \mathcal{L} \). Then, we define the exponential map \( \widetilde{\text{Exp}} \) and the logarithmic map \( \widetilde{\text{Log}} \) as in [40]

\[
\widetilde{\text{Exp}}_X(K) = [X] + [K] + \mathcal{D}(X) \exp \{ \mathcal{D}(K) \mathcal{D}(X)^{-1} \}, \quad (4)
\]

\[
\widetilde{\text{Log}}_K(X) = [X] - [K] + \mathcal{D}(K) \log \{ \mathcal{D}(K)^{-1} \mathcal{D}(X) \}. \quad (5)
\]

For the convenience of expression, we omit the subscript (i.e., \( \widetilde{\text{Exp}}_X = \text{Exp} \))

C. Fréchet Mean

The Fréchet mean [54] is an extension of the euclidean mean, widely used to aggregate representations in neural networks such as attention [4] and batch normalization [55], showing strong theoretical and practical interest in Riemannian data analysis. Given \( \{ P_i \}^N_{i=1} \) samples on \( S^+_d \), the Fréchet mean is defined as

\[
\mu = \arg \min_{G \in S^+_d} \sum_{i=1}^N \delta^2_R(G, P_i). \quad (6)
\]

where \( \delta_R \) is the Riemannian distance between two points on the manifold. Unfortunately, the solution to the minimization problem in (6) is unknown in closed-form, but is usually computed using an iterative solver, e.g., Karcher flow [56], that approximates the solution by computing in tangent space and mapping back to manifold space via logarithmic and exponential maps, respectively. However, the Fréchet mean in Cholesky space is closed and easy to compute. That is, an iterative operation is not required. We call this the log-Cholesky mean \( \mu_{\mathcal{L}} \) and it is defined as [40]

\[
\mu_{\mathcal{L}} = \frac{1}{N} \sum_{i=1}^N [X_i] + \exp \left\{ N^{-1} \sum_{i=1}^N \log \mathcal{D}(X_i) \right\}. \quad (7)
\]

D. Manifold ODE

We introduce the ordinary differential equations in manifold space. Suppose that \( z(t) \) is a differentiable curve with values in \( \mathcal{M} \). Then, by the definition of the tangent space, its derivative \( dz(t)/dt \in \mathcal{T}_{z(t)} \mathcal{M} \) \( \forall t \). This assumption leads to the following definition:

**Definition 1:** Let \( \mathcal{M} \) be a manifold space. A vector field \( f \) in regard to the dynamics is a \( C^1 \)-mapping such that

\[
\frac{dz(t)}{dt} = f(z(t), t) \in \mathcal{T}_{z(t)} \mathcal{M}. \quad (8)
\]

A vector field \( f \) is called a differential equation on the manifold, and a function \( z \) is called an integral curve or simply a solution of the equation.

According to Definition 1, a vector field is defined at arbitrary intervals in the manifold space. We consider the interval \( [t_1, t_2] \) such that \( z_1 = z(t_1) \). Since the solution of a differential equation depends on the initial point, we obtain the notation \( \varphi_{z_2}(z_1) = z(t_2) \) for the solution at time \( t_2 \) with an initial condition \( z_1 = z(t_1) \).

IV. CONTINUOUS MANIFOLD LEARNING ON RIEMANNIAN CHOLESKY SPACE

This section introduces our proposed method of continuous manifold learning in Cholesky space from sequential manifold-valued data for time-series modeling. We illustrate the overall procedures in Fig. 2.

A. Mapping to the Riemannian Cholesky Space

For manifold learning, we first transform the input time-series data into manifold space. Specifically, we exploit second-order statistics (i.e., covariance) of features represented via a function \( h_0 \), then use a shrinkage estimator [57], to estimate second-order statistics and thus, represent an SPD matrix in manifold space. Then, an SPD matrix \( S \) is decomposed into a lower triangular matrix \( X \) and its transpose \( X^\top \) by Cholesky decomposition \( C(S) = XX^\top \). We express the lower triangular matrix \( X \) as the sum of the strictly lower part and the diagonal part as \( X = [X] + \mathcal{D}(X) \).

Generally, it is challenging to train a network in SPD Riemannian manifold due to: (1) there are mathematical constraints in optimization, e.g., the convexity of weights in FM estimation [17], the validity of the output to be an SPD matrix [34], and (2) numerical errors that occur by distorting the geometric structure during the learning process, especially when the dimension of a matrix increases [58]. Meanwhile, when conducting operations in Cholesky space, due to the nice properties of the Cholesky matrices, we only consider the constraint that the elements of the diagonal matrix \( \mathcal{D}(X) \) should be positive, and the elements in \( [X] \) are unconstrained. Thus, by taking advantage of operating in the Cholesky space, it becomes relatively easier to solve an optimization problem.

B. Recurrent Network Model in Cholesky Space

An RNN equipped with gated recurrent units (GRU) [59] is successfully used for time-series modeling in various applications. Here, we introduce a novel Riemannian manifold GRU (RGRU) by reformulating gating operations in the Cholesky space. It is worth noting that the reformulated operations apply not only to GRU but also to LSTM. Compared with existing deep
learning methods for SPD matrix modeling [1], [17], [34], [35], [36], [41], [52], [53], as we formulate the optimization problem in a Cholesky space with a unique constraint of positivity in values of the diagonal part of $D(X)$, the optimization is solved relatively easily and with efficient computations, thus making their computation costs cheap.

Let us reformulate the gating operations and feature representations in terms of the Cholesky geometry with (i) weighted Fréchet mean ($\text{wFM}$), (ii) bias addition, and (iii) non-linearity. First, the wFM is defined using (7) for the weighted combination of Cholesky matrices as follows

$$\text{wFM}(\{X_i\}_{i=1}^{N}, w, \in \mathbb{R}_{\geq 0}^{N}) = \frac{1}{N} \sum_{i=1}^{N} (w_i \cdot [X_i]) + \exp \left\{ \sum_{i=1}^{N} w_i \cdot \log D(X_i) \right\},$$

where $\mathbb{R}_{\geq 0}$ denotes a space of non-negative real numbers and $w_i$ denotes the $i$th element of the weight vector $w$. Second, for the bias addition, we define an operator $\oplus$ on $\mathbb{L}_+$ [40] by

$$X \oplus Y = [X] + [Y] + D(X)D(Y).$$

Such an operation $\oplus$ is a smooth commutative group operation on the manifold $\mathbb{L}_+$. Since this group operation is a bi-invariant metric, we regard the operation for translation, analogous to the case in euclidean space. Third, regarding the non-linearity, since the off-diagonal values are unconstrained but the diagonal values should be positive, we handle elements of the off-diagonal and the diagonal parts separately by applying two independent activation functions accordingly and then adding the resulting values to obtain a valid Cholesky matrix, e.g.,

$$\text{tanh}([X]) + \text{softplus}(D(X)).$$

Given a sequence of SPD matrices $X = \{X_1, X_2, \ldots, X_T\}$ on $\mathbb{L}_+$ obtained from an input time-series sample by passing through the feature extractor $H_0$ and the covariance estimation module consecutively, we reformulate the gating and representation operations involved in our RGRU as follows

$$\begin{align*}
z_i &= \sigma(\text{wFM}(\{X_i, H_{i-1}\}, W_z) \oplus B_z), \\
r_i &= \sigma(\text{wFM}(\{X_i, H_{i-1}\}, W_r) \oplus B_r), \\
I_i &= \text{wFM}(\{X_i, r_i \oplus H_{i-1}\}, W_i) \oplus B_i, \\
H_i &= \text{tanh}(I_i) + \text{softplus}(D(I_i)), \\
H_{i+1} &= (1 - z_i) \cdot H_{i+1} + z_i \cdot H_i,
\end{align*}$$

where $H_i \in \mathbb{L}_+$ is the hidden state in RGRU, $z_i \in \mathbb{L}_+$, $r_i \in \mathbb{L}_+$ and $H_i \in \mathbb{L}_+$ are the update gate, the reset gate, and candidate hidden state, respectively; $\{W_z \in \mathbb{L}_+, W_r \in \mathbb{L}_+, W_i \in \mathbb{L}_+\}$ are learnable weights and $\{B_z \in \mathbb{L}_+, B_r \in \mathbb{L}_+, B_i \in \mathbb{L}_+\}$ are corresponding learnable biases; $\sigma(\cdot)$ is a logistic sigmoid function, $\text{tanh}(\cdot)$ is a hyperbolic tangent function, and $\text{softplus}(\cdot)$ is a softplus function, and $\oplus$ denotes an operator of element-wise multiplication that satisfies the group axiom. It is worth noting that without loss of generality, any activation function with a positive output range, e.g., sigmoid, can be used in the position of a softplus function.

C. Neural Manifold ODE

For continuous manifold learning, we further leverage the technique of neural manifold ODEs [18] in sequential data modeling and model the dynamics of $f$ by a neural network with its learnable parameters $\Theta$. Here, we define both the forward and backward pass gradient computations for network parameter learning. Notably, the forward pass is computed in manifold
**Algorithm 1: Pseudo Algorithm of ODE-RGRU.**

**Input:** Time-series data $S = \{S_i\}_{i=1,...,T}$ and time stamps $t = \{t_i\}_{i=1,...,T}$, learnable model parameters $\theta$, $\Theta$

**Output:** $\{H_i\}_{i=1,...,T}$

1: $\{S'_i\}_{i=1,...,T} = h_0(S)$
2: $\{C'_i \in S'_d\}_{i=1,...,T} = SE(\{S'_i\}_{i=1,...,T})$ //SE:
   3: \text{Shrinkage Estimation}
   4: \text{CD: Cholesky Decomposition}
5: $H_0 = I$ //Identity matrix
6: for $i$ in $1, 2, ..., T$ do
7: \ $H'_i = \text{ODESolve}(f_\Theta, \text{Log}(H_{i-1}), (t_{i-1}, t_i))$
8: \ $H_i = \text{RGRU}(\text{Exp}(H'_i), X_i)$ // (11)
9: \ return $\{H_i\}_{i=1,...,T}$

space but the backward pass is defined solely through an ODE in euclidean space.

1) **Forward Mode:** Forward mode integration is broadly classified into two groups: a projection method [60] and an implicit method [18], [60], [61]. In this work, we employ an implicit method due to its general applicability in any manifold space and its definition using step-based solvers [61], [62]. Particularly, we use a variant of an Euler method solver with an update step $h_{i+\epsilon} = \exp_{H_i}(\epsilon f_\Theta(H_i, t))$ for the Riemannian exponential map [62]. Here, $\epsilon$ is a time step.

2) **Backward Mode:** Recently, [42] and [18] proposed an adjoint sensitivity method to effectively compute the gradients and calculate the derivative of a manifold ODE, independently. In our method, we use the adjoint method to compute the derivative of a manifold ODE.

In differential geometry, $D_{x,t} f_\Theta : T_x M \to T_{z(t)} N$ defines a derivative of a function $f_\Theta : M \to N$ mapping between two manifolds.

**Theorem 1:** [18] We define a loss function $E : M \to \mathbb{R}$. Suppose there is an embedding of manifold space $M$ in euclidean space $\mathbb{R}^d$. Let the adjoint state be $a(t) = D_{x(t)} E$, then the adjoint satisfies

$$\frac{da(t)}{dt} = -a(t) D_{z(t)} f_\Theta(z(t), t)$$ (12)

In the implementation, we construct the differential equation

$$\frac{d\text{Log}(H_i)}{dt} = D_{\text{Exp}(H_i)} \text{Log}(f_\Theta(\text{Exp}(H_i), t))$$ (13)

Then, $\text{Log}(H_i)$ is solved using the numerical integration technique. Finally, we can update $H_i$ to $H_{i+\epsilon}$. With the adjoint state, we calculate the gradients with respect to the start and end time points, $t_s$, $t_e$ with the initial condition $H_s$, and the weights $\Theta$.

**D. ODE-RGRU**

Here, we describe the systematic integration of manifold ODE and RGRU in a unified framework for sequential continuous manifold modeling, called ODE-RGRU. Formally, the ODE-RGRU is formulated as

$$H'_i = \text{ODESolve}(f_\Theta, \text{Log}(H_{i-1}), (t_{i-1}, t_i))$$ (14)

$$H_i = \text{RGRU}(\text{Exp}(H'_i), X_i),$$ (15)

where $X_i \in \mathcal{L}_+$ is an input manifold-valued feature and $H_i$ is a latent state at the $i$-th timepoint, for $i = 1, \ldots, T$. We define such a hidden state $H'_i$ to be the solution on tangent space after mapping via $\text{Log}$ to an ODESolver and then the output of the ODESolver is orthogonally projected to a manifold space by an exponential map $\text{Exp}$. The initial hidden state $H_0$ is set to identity matrices $I$ that satisfy the SPD constraint. For each observation over time, the corresponding hidden state is updated in the Cholesky space using (11), i.e., RGRU. The overall algorithm of ODE-RGRU is summarized in Algorithm 1. Note that Algorithm 1 focused on the representation of the hidden state, while the proposed model concentrated on mapping to the Cholesky space within the SPD matrix. However, considering the diffeomorphic properties between these two spaces, it is also possible to apply a mapping from the Cholesky space to the SPD space. Thus, depending on the desired objective, mapping between the Cholesky space and the SPD space is feasible, offering flexibility in achieving the intended goals.

**E. Computational Time Efficiency**

It should be noted that the computation of manifold space point representations in our proposed model relies on CPU-based operations, which may result in slower training times compared to conventional time series modeling approaches. However, this limitation is mitigated by the inherent statistical characteristics of data and the model’s capacity for effective generalization. Consequently, despite the potential trade-off in computational speed, the model enables efficient optimization and exhibits rapid convergence due to its ability to leverage the intrinsic statistical properties of data [63].

To learn geometric information in a neural network through manifold learning, the internal operations should consider and satisfy manifold constraints. Specifically, the SPD matrix appears in various domains, but it is difficult to train a neural network because of rigid constraints, i.e., symmetries and all positive eigenvalues. To tackle such challenges, [34] train their model in a compact Stiefel manifold space and [41] applied euclidean operations in a tangent space with a logarithmic map and switched to manifold space with an exponential map. Most operations are performed through the Eigenvalue decomposition with a computational complexity of $O(k \cdot n^3)$ on $k$ iterations for Fréchet mean estimation. That is, as the dimensions of the matrices of interest increase, the computational cost increases exponentially. Therefore, the previous work [64] derives explicit gradient expressions so that manifold learning could be performed in a reasonable computational time, but it was limited to hyperbolic space. Consequently, it remains that a computational problem exists in the Riemannian manifold for the Fréchet mean estimation, thus making the model’s training slow and unstable.
In this work, we propose to exploit the Cholesky decomposition, which finds a unique lower triangular matrix for an SPD matrix. The nice properties of a Cholesky matrix allow efficient numerical calculations. Moreover, it has simple and easy-to-compute expressions for Riemannian exponential and logarithmic maps. Notably, it provides a closed-form solution for the weighted Fréchet mean with a linear computational complexity.

V. EXPERIMENTS

In this section, we evaluate the proposed method for real-world applications of video action recognition, sleep staging classification, and various multivariate time-series modeling tasks by using publicly available datasets. Our implementation code used for the experiments is available at “https://github.com/ku-milab/Deep-Efficient-Continuous-Manifold-Learning” and all the implementation details are provided in our GitHub repository.

A. Datasets and Preprocessing

1) Action Recognition: For the action recognition task, we used the UCF11 [65] dataset. The UCF11 dataset is composed of 1,600 video clips, with 11 action categories, e.g., basketball shooting, biking/cycling, diving, golf swinging, etc. The length of the video frames varies from 204 to 1492 per clip with a resolution of each frame being 320 × 240. Similar to previous work [1], [17], we downsized all frames to 160 × 120 and sampled 50 frames from each clip, i.e., N = 50, with an equal time gap between frames. Then, we performed the 5-fold cross-validation for fair evaluation, while keeping the class balance.

2) Sleep Staging Classification: We used the SleepEDF-20 [66] dataset for the sleep staging classification task. The SleepEDF-20 is the Sleep Cassette that comprises 20 subjects (10 males and 10 females) aged 25-34. Two polysomnographies were recorded during day-night periods, except for subject 13 who lost the second night due to a device problem. All signals were segmented into lengths of 30 seconds and manually labeled into 8 categories, ‘wake,’ ‘rapid eye movement (REM),’ ‘non REM1/2/3/4,’ ‘movement,’ and ‘unknown,’ by sleep experts [67]. In this work, similar to the previous studies [5], [68], [69], [70], we merged ‘non REM3’ and ‘non REM4’ stages into a ‘non REM3’ stage and removed samples labeled as ‘movement’ and ‘unknown.’ Here, we used spectrogram data rather than raw signals [71]. To achieve this, we sampled all signals at 100 Hz rate and applied short-time Fourier transformation to compute spectrograms. For this experiment, we used single-channel EEG and 2-channel EEG/EOG samples. Note that we did not exploit electromyogram recordings due to their incompleteness. As a result, we obtained a multi-channel signal S ∈ R^{F × T × C} where F, T, and C denote the number of frequency bins, the number of spectral columns, and the number of channels, respectively. We set F = 129, T = 29, C = 1 or C = 2 (for the single-channel EEG or two-channel EEG/EOG). For the staging classification, 10 continuous segments of data were used as input, i.e., N = 10. For a fair evaluation, we conducted a 20-fold cross-validation by maintaining a class balance similar to [5], [68], [69], [70].

3) Multivariate Time Series Classification: To evaluate various sensor data in the classification task not only for sleep staging classification, we used the UEA archive [44], a well-known multivariate time-series data. The UEA archive consists of 30 real-world multivariate time-series data for various classification applications, e.g., Human Action Recognition, Motion classification, etc. The dimension of the archive range from 2 dimensions in AtrialFibrillation to 1345 in DuckDuckGeese, and the time length is from 8 in PenDigits to 17984 in EigenWorms. The dataset is divided into train and test sets with the train size ranging from 12 to 30000. Detailed information about each dataset can be found in [44].

4) Time Series Forecasting & Imputation: For the forecasting task, we used four public datasets, including three ETT datasets [23] and the Electricity dataset [72]. The ETT is the dataset that collects electricity transformer temperature, which is an important indicator in power long-term deployment. The ETT dataset consists of ETT1h, ETT2h at a 1-hour level and ETT15m at a 15-minute level, and each point is expressed as an oil temperature and 6 power load features. Electricity dataset is the measurement of electric power consumption that contains 2,075,259 measurements.

For the imputation task, we collected data from the TADPOLE database,1 which is based on the Alzheimer’s Disease Neuroimaging Initiative (ADNI) cohort. This dataset comprises information from 1,737 patients and encompasses 1,500 biomarkers obtained over 12,741 visits spanning 22 time periods. Although the TADPOLE database offers numerous biomarkers for predicting the Alzheimer’s disease (AD) spectrum, our study focused on six volumetric MRI features, T1-weighted MRI scans, and cognitive test scores, such as the mini-mental state exam (MMSE), Alzheimer’s disease assessment scale (ADAS)-cog11, and ADAS-cog13.

B. Experimental Setting

1) Action Recognition: In our work, we compared the proposed method with state-of-the-art manifold learning methods and typical deep learning models in the action recognition task. For the baseline methods representing data in euclidean space, we exploited GRU [59], LSTM [75], TT-GRU, and TT-LSTM [20]. For GRU and LSTM, we first extracted spatial features using 3 convolutional layers with a kernel size of 7 × 7 and channel dimensions of 10, 15, and 25, respectively. Further, batch normalization [55], leaky rectified linear units (LeakyReLU) [76], and max pooling were followed after each convolutional layer. Then, the extracted spatial features were flattened and fed into the recurrent models, i.e., GRU and LSTM, both of which had 50 hidden nodes. For TT-GRU and TT-LSTM, we reshaped an input sample to 8 × 20 × 20 × 18 and set the rank size to 1 × 4 × 4 × 4 × 1. Finally, we obtained the 4 × 4 × 4 × 4 output shape.

For the other baseline methods considered in the action recognition experiments, we employed SPDSRU [17] and ManifoldDCNN [1]. For covariance matrices used in SPDSRU

1https://tadpole.grand-challenge.org/Data
and ManifoldDCNN, we adopted a covariance block analogous to [77] to get the reported testing accuracies. Then, for SPDSRU, 2 convolutional layers with $7 \times 7$ kernels, 7 channels for the final output were chosen, therefore the dimension of the covariance matrix was $8 \times 8$ the same as in [17]. Finally, based on [1], ManifoldDCNN also used 2 convolutional layers with $7 \times 7$ kernels and set 6 channels for the final output, i.e., the dimension covariance matrix was $7 \times 7$.

While training our proposed method, we chose 3 convolutional layers with a kernel size of $7 \times 7$ and exploited batch normalization [55], LeakyReLU activation [76], and max pooling for $h_{\theta}$. Further, we set the number of output dimensions to 32, i.e., the dimension of the SPD matrix was $32 \times 32$. Therefore, the diagonal matrix and strictly lower triangular matrix through Cholesky decomposition were expressed as a 32-dimensional vector and a vector of $32 \times (32 - 1)/2$ dimension, respectively. Moreover, similar to neural ODEs [42], we employed a multi-layer perceptron (MLP) with a tanh activation of $f_{\theta}$. Finally, we set the number of hidden units to 100 for RGRU.

2) Sleep Staging Classification: For the sleep staging classification experiments, we compared our proposed method with the state-of-the-art methods of XSleepNet [5], SeqSleepNet [71], FCNN+RNN [5], DeepSleepNet [70], U-time [73], and IITNet [74]. We report the performance of those competing methods by taking from [5].

By following [5], we used a filterbank layer [78] for time-frequency input data. Through the learnable filterbank layer, the spectral dimension of the input data was reduced from $F = 129$ to 32. Here, 2 convolutional layers followed by batch normalization and a LeakyReLU activation were used for $h_{\theta}$. The output dimension was set as 8, i.e., the dimension of the SPD matrix was $8 \times 8$. We also used an MLP with a tanh for $f_{\theta}$ and set 50 hidden units for RGRU. Finally, since the sleep staging of a sample is related to neighboring samples, we also exploited a bidirectional structure in RGRU to consider such useful information, the same as in other competing methods.

3) Multivariate Time Series Classification: In the UEA archive experiment, we compared our proposed method with eight different methods, including three benchmarks [44], a bag-of-pattern based approach [8], and deep learning-based methods [9], [10], [11], [21], [22], [24]. The three benchmarks are euclidean Distance (EDI), which is a 1-nearest neighbor with covariance matrix was expressed as a 32-dimensional vector and a vector of $32 \times (32 - 1)/2$ dimension, respectively. Moreover, similar to neural ODEs [42], we employed a multi-layer perceptron (MLP) with a tanh activation of $f_{\theta}$. Finally, we set the number of hidden units to 100 for RGRU.

5) Training Details: For all experiments except forecasting and imputation, we used a multi-class cross-entropy loss and employed an Adam optimizer with a learning rate of $10^{-4}$ and an $\ell_2$ regularizer with a weighting coefficient of $10^{-3}$. We trained our proposed network for 1,000 (action recognition), 20 (sleep staging classification) epochs, and 400 (UEA archive, ETT, Electricity, and ADNI) iterations with a batch size of 32. For forecasting and imputation, we used a mean squared error (MSE) loss. For the learnable parameters of RGRU, which should be in Cholesky space, i.e., $W_{z}, W_{r}, W_{l}, B_{z}, B_{r}, B_{l} \in \mathbb{C}^+$, we only needed to care about the diagonal elements. In order for that, we simply applied an absolute operator during backpropagation.

C. Experimental Results and Analysis

1) Classification Performance: Action Recognition: As presented in Table III, the proposed method outperformed the competing baseline methods with a large margin (7.1% $\sim$ 18.0%) in the action recognition task. Notably, the proposed method achieved the best performance although the number of its tunable parameters was significantly smaller than the other euclidean representation-based methods. Based on these promising results, we presume that the proposed method could learn geometric features in complex actions efficiently.

Sleep Staging Classification: We evaluated the methods in terms of accuracy (Acc), Cohen’s kappa value (\kappa) [82], macro F1-score (MF1) [83], averaged sensitivity (Sen), and averaged
To handle irregularly-sampled time-series data, we evaluated the method for the statistical validity of our proposed method. As shown in 10 datasets, and in particular, Ering and InsectWingbeat showed best performance among the competing methods. Also, compared with distance-based methods such as NS [22], TapNet [9], and ShapeNet [10], it showed sufficiently high performance even on small-sized datasets (e.g., AtrialFibrillation, BasicMotions). This shows that our proposed method learns the trajectory of time well to discover a sufficient representation even with limited training samples. Our proposed method ranked the top on average among the competing methods and previous works with compatible results.

**Multivariate Time Series Classification**: The experimental results from the UEA archive are shown in Table V. For each dataset, we evaluated the accuracy and computed the total best accuracy, average rank, and Wilcoxon-signed rank test by following [10]. Our proposed method showed the best performance in 10 datasets, and in particular, Ering and InsectWingbeat showed overwhelming performance compared to the competing baseline methods. Also, compared with distance-based methods such as NS [22], TapNet [9], and ShapeNet [10], it showed sufficiently high performance even on small-sized datasets (e.g., AtrialFibrillation, BasicMotions). This shows that our proposed method learns the trajectory of time well to discover a sufficient representation even with limited training samples. Our method ranked on top average among the competing methods considered in this work by showing the highest performance over 10 datasets in general time-series classification tasks. We also conducted a Wilcoxon-signed rank test following in [10], [84], [85] for the statistical validity of our proposed method. As presented in the last row of Table V, all results are statistically significant, except for WEASEL+MUSE and ShapeNet.

Compared to WEASEL+MUSE, we observed that ODE-RGRU consistently outperforms, particularly in scenarios involving high-dimensional or large datasets such as InsectWingbeat and PEMS-SF. Conversely, WEASEL+MUSE often grappled with challenges, including memory issues [9], which hindered its execution. The limitation in WEASEL+MUSE can be attributed to the necessity of generating symbols (words) for each sub-sequence, considering its length and dimension. Notably, in the case of high-dimensional time series data, the size of the symbol dictionary can significantly increase, making the approach less suitable for efficiently handling large datasets.

ShapeNet has empirically exhibited performance levels comparable to our proposed method, but we argue it has certain inherent limitations. Specifically, ShapeNet requires the identification of shapelets tailored to each dataset. This procedure can be time-consuming and might not fully encompass crucial details regarding variables and subsequence positions. In contrast, our proposed approach addresses these limitations by acquiring universal representations encompassing essential temporal and geometric information within time series data. This methodological disparity renders our approach versatile and adaptable across various tasks, as it prioritizes the capture of general time series characteristics.

2) **Forecasting & Imputation Performance**: We evaluated the competing methods in terms of MSE and reported the results in Table VIII. Our proposed method demonstrated the best performance in 8 cases and exhibited comparable results in most instances. However, the proposed method generally showed competitive performance in long-term predictions but tended to show somewhat poor performance in short-term predictions.

This discrepancy might be attributed to using a sliding window for conducting prediction tasks on long-time series data. While this approach extracts temporal and geometric information into the manifold space, it introduces challenges in incorporating semantic information into the prediction process. Although TS2Vec [11], which learned semantic level time-series representation, showed better performance overall, our proposed method showed plausible performance in the ETTh1 and electricity datasets.

We also conducted cognitive score prediction to demonstrate the imputation ability of our method. We evaluated the metrics of mean absolute percentage error (MAPE) and coefficient of determination ($R^2$). In addition, we performed a statistical significance test using the Wilcoxon signed-rank test to compare the performance of our framework with other comparative methods. The results, as presented in Table VII, demonstrated that our proposed method achieved the highest MAPE and $R^2$ for all features. This outcome indicates the superior performance of our approach compared to the other methods under consideration.

3) **Irregularly-Sampled Data (CharacterTrajectories)**: To show the effectiveness of our model for continuous models [12], [13], [16] dealing with irregularly-sampled time-series data, we conducted additional experiments using a CharacterTrajectories dataset from the UEA archive. To do this, we randomly dropped 30%, 50%, and 70% of time-series data across channels by following [16]. While training our proposed model with irregular settings, we set the model structure to be the same as with regular settings. However, it was difficult to use the time window due to random missing values, so we calculated the SPD matrix for each time point using feature maps obtained via 3 convolutional layers with a kernel size of 1 and batch normalization, LeakyReLU activation.

We evaluated the mean and variance of accuracy through five repeats and reported in Table VI. Our proposed method showed the same or higher performance at 30% and 70% drops
compared with the baseline methods. Although it showed 0.5 lower performance in 50% dropped than Neural CDE [16], it was still an overwhelming performance compared to other continuous models [12], [13]. This indicates that our proposed model encouraged sufficient improvement even on irregularly-sampled data.

4) Ablation Study: To verify the effectiveness of the manifold ODE in our method, we conducted ablation studies by taking off the manifold ODEs. We conducted experiments on the SleepEDF-20 dataset, and the results are illustrated in Table IV. Without manifold ODEs, ODE-RGRU (w/o ODEFunc.) showed a lower performance than ODE-RGRU. In particular, MFI and sensitivity were the lowest among baseline models in both cases of EEG and EEG/EOG. This shows that time-series modeling has made a sufficient improvement in learning geometric characteristic representations through manifold ODE.

We computed the accuracy (Acc), Cohen’s kappa value (κ), macro F1-score (MFI), averaged sensitivity (Sen), and averaged specificity (Spec).

We highlight in bold where ODE-RGRU’s performances are better or equal to that of all the baseline methods and previous works with comparable results.

### Table VI

| Dataset | ArticulatoryWordRecognition | Articulation | BasicMotion | CharacterTrajectories | Cricket | DuckDuckGoose | EigenWorms | Epilepsy | ERing | EthanolConcentration | FaceDetection | FingerMovements | Handwriting | Heartbeat | InsectWingbeat | JapaneseVerbal | Libras | LSST | MotorImagery | NATOPS | PEMS-SF | PenDigits | Phone | RegNet | RacketSports | SelfRegulationSC1 | SelfRegulationSC2 | SpokenArabicDigits | StandWalkJump | UWaveGestureLibrary |
|---------|-----------------------------|-------------|-------------|-----------------------|--------|---------------|------------|-----------|-------|----------------------|--------------|------------------|-------------|-----------|----------------|--------------|--------|------|-------------|--------|--------|----------|------|--------|-------------|-----------------|-----------------|------------------|-------------|------------------|
| Acc     | 88.9 ± 0.2                 | 98.7 ± 0.2  | 98.8 ± 0.2  | 98.6 ± 0.4            | 94.6 ± 0.6 | 94.7 ± 0.9   | 93.0 ± 0.7  | 93.4 ± 0.7  | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 |
| MFI     | 98.6 ± 0.3                 | 98.6 ± 0.3  | 98.6 ± 0.3  | 98.6 ± 0.3            | 95.3 ± 0.6 | 95.3 ± 0.6   | 95.3 ± 0.6  | 95.3 ± 0.6  | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 |
| Sen     | 98.6 ± 0.3                 | 98.6 ± 0.3  | 98.6 ± 0.3  | 98.6 ± 0.3            | 95.3 ± 0.6 | 95.3 ± 0.6   | 95.3 ± 0.6  | 95.3 ± 0.6  | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 |
| Spec    | 98.6 ± 0.3                 | 98.6 ± 0.3  | 98.6 ± 0.3  | 98.6 ± 0.3            | 95.3 ± 0.6 | 95.3 ± 0.6   | 95.3 ± 0.6  | 95.3 ± 0.6  | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 |

We computed the accuracy of all comparison methods, total best accuracy, average rank, and Wilcoxon-signed rank test. We highlight in bold where ODE-RGRU’s performances are better or equal to that of all the baseline methods and previous works with comparable results.

### Table VII

| Method | Test Accuracy | 50% dropped | 30% dropped | 70% dropped | 50% dropped | 30% dropped | 70% dropped |
|--------|---------------|-------------|-------------|-------------|-------------|-------------|-------------|
| ODE-RGRU (Ours) | 98.9 ± 0.2 | 98.3 ± 0.2 | 98.6 ± 0.3 | 98.6 ± 0.3 | 98.6 ± 0.3 | 98.6 ± 0.3 | 98.6 ± 0.3 |
| Neural CDE [16] | 98.7 ± 0.5 | 98.6 ± 0.2 | 98.6 ± 0.4 | 98.6 ± 0.4 | 98.6 ± 0.4 | 98.6 ± 0.4 | 98.6 ± 0.4 |
| ODE-RNN [12] | 95.4 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 | 95.3 ± 0.6 |
| GRU-DR [17] | 94.2 ± 2.1 | 90.2 ± 4.6 | 92.3 ± 3.6 | 92.3 ± 3.7 | 92.3 ± 3.7 | 92.3 ± 3.7 | 92.3 ± 3.7 |

We computed the mean and variance of accuracy on five repeats. We highlight in bold where ODE-RGRU’s performances are better or equal to that of all the baseline methods and previous works with comparable results.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
TABLE VII

| Method      | MMSE   | ADAS-cog11 | ADAS-cog13 |
|-------------|--------|------------|------------|
|             | MAPE   | $R^2$      | MAPE       | $R^2$      | MAPE       | $R^2$      |
| LSTM-M      | 0.173±0.030* | -0.412±1.143* | 0.529±0.433* | 0.321±0.173* | 0.863±0.289* | 0.302±0.267* |
| LSTM-F      | 0.235±0.110* | -0.553±0.495* | 0.629±0.353* | 0.196±0.494* | 0.790±0.152* | 0.177±0.468* |
| MRNN [80]   | 0.149±0.031* | 0.168±0.284* | 0.930±0.224* | 0.262±0.184* | 0.920±0.234* | 0.263±0.187* |
| MinimalRNN [81] | 0.175±0.052* | 0.472±0.115* | 0.565±0.142* | 0.569±0.038* | 0.431±0.111 | 0.635±0.049* |
| Ours        | 0.099±0.018 | 0.608±0.067 | 0.441±0.043 | 0.689±0.036 | 0.403±0.043 | 0.726±0.034 |

We highlight in bold where ODE-RGRU’s performances are better or equal to that of all the baseline methods and previous works with compatible results.

TABLE VIII

| Dataset  | $H$ | TCN [29] | Informer [23] | TS2Vec [11] | ODE-RGRU (Ours) |
|----------|-----|----------|----------------|-------------|-----------------|
| ETTb1    | 24  | 0.767    | 0.577          | 0.599       | 0.643           |
|          | 48  | 0.713    | 0.685          | 0.629       | 0.616           |
|          | 168 | 0.995    | 0.931          | 0.755       | 0.838           |
|          | 336 | 1.175    | 1.128          | 0.907       | 0.885           |
|          | 720 | 1.453    | 1.215          | 1.048       | 0.862           |
| ETTb2    | 24  | 1.365    | 0.720          | 0.398       | 0.781           |
|          | 48  | 1.395    | 1.457          | 0.580       | 0.881           |
|          | 168 | 3.166    | 3.489          | 1.901       | 3.945           |
|          | 336 | 3.256    | 2.723          | 2.304       | 2.472           |
|          | 720 | 3.690    | 3.467          | 2.650       | 2.283           |
| ETTh1    | 24  | 0.324    | 0.323          | 0.443       | 0.672           |
|          | 48  | 0.477    | 0.494          | 0.582       | 0.783           |
|          | 96  | 0.636    | 0.678          | 0.622       | 0.648           |
|          | 288 | 1.270    | 1.056          | 0.709       | 0.755           |
|          | 672 | 1.381    | 1.192          | 0.786       | 0.835           |
| ETTh2    | 24  | 0.305    | 0.312          | 0.287       | 0.260           |
|          | 48  | 0.517    | 0.392          | 0.307       | 0.396           |
|          | 96  | 0.358    | 0.515          | 0.332       | 0.328           |
|          | 288 | 0.349    | 0.759          | 0.349       | 0.336           |
|          | 672 | 0.447    | 0.969          | 0.375       | 0.375           |

Total best MSE: 1 1 10 8

We computed the mean squared error (MSE) of all comparison methods. We highlight in bold where ODE-RGRU’s performances are better or equal to that of all the baseline methods and previous works with compatible results.

5) Training Speed: Our method uses Cholesky decomposition to map the SPD manifolds into Cholesky space. The Cholesky decomposition greatly reduces the computational costs, allowing us to use the larger SPD matrix than the existing geometric deep-learning methods [34], [52]. Therefore, by using the large SPD matrix, the proposed method could learn various features of the large dimensional input data. We compared the learning curves of three comparative manifold deep-learning methods in Fig. 3. Remarkably, our proposed method showed its convergence faster and more stable than the others.

VI. CONCLUSION

Manifold learning with deep neural networks for SPD matrices is a challenge due to their hard constraints. In this paper, we proposed a novel deep efficient continuous manifold neural network, called ODE-RGRU, by systematically combining GRU and ODE in Cholesky space, and efficiently resolving difficulties related to computational costs in optimization. Specifically, we first reformulated the gating operations in GRU, thus making it work in the Riemannian manifold. Furthermore, to learn continuous geometric feature representations from sequential SPD-valued data, we additionally devised and applied a manifold ODE to the recurrent model. In our extensive experiments on various time-series tasks, we empirically demonstrated its effectiveness and suitability for time-series representation modeling, by achieving state-of-the-art performance for classification, forecasting, and imputation tasks.

REFERENCES

[1] X. Zhen, R. Chakraborty, N. Vogt, B. B. Bendlin, and V. Singh, “Dilated convolutional neural networks for sequential manifold-valued data,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 10620–10630.
[2] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, “FairMOT: On the fairness of detection and re-identification in multiple object tracking,” Int. J. Comput. Vis., vol. 129, no. 11, pp. 3069–3087, 2021.
[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” 2018, arXiv: 1810.04805.
[4] A. Vaswani et al., “Attention is all you need,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 5998–6008.
[5] H. Phan, O. Y. Chen, M. C. Tran, P. Koch, A. Mertins, and M. De Vos, “XSleepNet: Multi-view sequential model for automatic sleep staging,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 9, pp. 5903–5915, Sep. 2022.
[6] W. Ko, E. Jeon, S. Jeong, and H.-I. Suk, “Multi-scale neural network for EEG representation learning in BCI,” IEEE Comput. Intell. Mag., vol. 16, no. 2, pp. 31–45, May 2021.
[7] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, “Recurrent neural networks for multivariate time series with missing values,” Sci. Rep., vol. 8, no. 1, pp. 1–12, 2018.
[8] P. Schäfer and U. Leser, “Multivariate time series classification with WEASEL MUSE,” 2017, arXiv: 1711.11343.
[38] C. Chefd'Hôtel, R. Tschumperlé, and O. Faugeras, “Regularizing flows for constrained matrix-valued images,” J. Math. Imaging Vis., vol. 20, no. 1, pp. 147–162, 2004.

[39] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, “Geometric means in a novel vector space structure on symmetric positive-definite matrices,” SIAM J. Matrix Anal. Appl., vol. 29, no. 1, pp. 328–347, 2007.

[40] Z. Lin, “Riemannian geometry of symmetric positive definite matrices via cholesky decomposition,” SIAM J. Matrix Anal. Appl., vol. 40, no. 4, pp. 1353–1370, 2019.

[41] Z. Gao, Y. Wu, Y. Jia, and M. Harandi, “Learning to optimize on SPD manifolds,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 7790–7799.

[42] R. T. Chen, Y. Rubanova, J. Bettencourt, and K. D. Duvenaud, “Neural ordinary differential equations,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 6572–6583.

[43] D. Rezende and S. Mohamed, “Variational inference with normalizing flows,” in Proc. 32nd Int. Conf. Mach. Learn., 2015, pp. 1530–1538.

[44] A. Bagnall et al., “The UEA multivariate time series classification archive,” 2018, arXiv:1811.00977.

[45] J. Lines and A. Bagnall, “Time series classification with ensembles of elastic distance measures,” Data Mining Knowl. Discov., vol. 29, no. 3, pp. 565–592, 2015.

[46] A. P. Ruiz, M. Flynn, J. Large, M. Middlehurst, and A. Bagnall, “The great multivariate time series classification bake off: A review and experimental evaluation of recent algorithmic advances,” Data Mining Knowl. Discov., vol. 35, no. 2, pp. 401–449, 2021.

[47] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 7132–7141.

[48] A. V. D. Oord et al., “WaveNet: A generative model for raw audio,” 2016, arXiv:1609.03499.

[49] F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A unified embedding for face recognition and clustering,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2015, pp. 815–823.

[50] Y.-H. Chen and J.-T. Chien, “Continuous-time attention for sequential modeling,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2020, pp. 11042–11052.

[51] M. Fréchet, “Les éléments aléatoires de nature quelconque dans un espace dérivé,” Annales de l’institut Henri Poincaré, vol. 10, no. 4, 1948, pp. 215–310.

[52] S. Joffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in Proc. 32nd Int. Conf. Mach. Learn., 2015, pp. 448–456.

[53] H. Karcher, “Riemannian center of mass and mollifier smoothing,” Commun. Pure Appl. Math., vol. 30, no. 5, pp. 509–541, 1977.

[54] Y.-Y. Chen and J.-T. Chien, “Continuous-time attention for sequential learning,” in Proc. AAAI Conf. Artif. Intell., 2021, pp. 7116–7124.

[55] I. D. Jordan, P. A. Sokol, and I. M. Park, “Gated recurrent units viewed as continuous-time dynamical systems,” Front. Comput. Neurosci., vol. 15, 2021, Art. no. 678158.

[56] C. Yildiz, M. Heinonen, and H. Lahdesmaki, “OD2VEAE: Deep generative second order ODEs with Bayesian neural networks,” in Proc. Adv. Neural Inf. Process. Syst., 2019, pp. 13434–13443.

[57] F. Kadir, J. Morrill, J. Foster, and T. Lyons, “Neural controlled differential equations for irregular time series,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 6696–6707.

[58] R. Chakraborty et al., “A statistical recurrent model on the manifold of symmetric positive definitemetrics,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 8897–8908.

[59] A. Lou et al., “Neural manifold ordinary differential equations,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 17548–17558.

[60] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, “The graph neural network model,” IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 10, pp. 5016–5029, Oct. 2010.

[61] Z. Yue et al., “TS2Vec: Towards universal representation of time series,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 2114–2124.

[62] A. Kendall and R. Cipolla, “Geometric loss functions for camera pose regression with deep learning,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 6555–6564.

[63] X. Huang and L. Van Gool, “A Riemannian network for SPD matrix learning,” in Proc. IEEE Conf. Artif. Intell., 2017, pp. 2036–2042.

[64] Y.-J. Suh and B. H. Kim, “Riemannian embedding banks for common spatial patterns with EEG-based SPD neural networks,” in Proc. AAAI Conf. Artif. Intell., 2021, pp. 854–862.

[65] R. Chakraborty, J. Bouza, J. Manton, and B. C. Venuri, “ManifoldNet: A deep neural network for manifold-valued data with applications,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 2, pp. 799–810, Feb. 2022.

[66] X. S. Nguyen, L. Brun, O. Lézoray, and S. Bougleux, “A neural network based on SPD manifold learning for skeleton-based hand gesture recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 12036–12045.

[67] C. ChefdeHotel, T. SchumpeRDeriche, and O. Faugeras, “Regularizing flows for constrained matrix-valued images,” J. Math. Imag. Vis., vol. 20, no. 1, pp. 147–162, 2004.

[68] F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A unified embedding for face recognition and clustering,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2015, pp. 815–823.
Seungwoo Jeong (Student Member, IEEE) received the BS degree in Mathematics and Statistics from Hankuk University of Foreign Studies, Yongin, South Korea, in 2019. He is currently working toward the PhD degree with the Department of Artificial Intelligence, Korea University, Seoul, South Korea. His current research interests include time-series analysis, brain-computer interface, and machine/deep learning.

Wonjun Ko received the PhD degree in brain and cognitive engineering from Korea University, Seoul, Republic of Korea, in 2022. He was a Data Scientist with SK Hynix Inc., Incheon, Republic of Korea, from 2022 to 2023. He is currently an assistant professor with the School of AI Convergence, Sungshin Woman’s University, Seoul, Republic of Korea. His current research interests include machine learning, representation learning, and data mining.

Ahmad Wisnu Mulyadi received the bachelor’s degree in computer science education from the Indonesia University of Education, Bandung, Indonesia, in 2010. He is currently working toward the PhD degree with the Department of Brain and Cognitive Engineering, Korea University, Seoul, South Korea. His current research interests include machine/deep learning in healthcare, biomedical image analysis, and graph representation learning.

Heung-Il Suk (Senior Member, IEEE) is currently a professor with the Department of Artificial Intelligence and an adjunct professor at the Department of Brain and Cognitive Engineering with Korea University. He was a visiting professor with the Department of Radiology, Duke University between 2022 and 2023. He was awarded a Kakao Faculty Fellowship from Kakao and a Young Researcher Award from the Korean Society for Human Brain Mapping (KHBM), in 2018 and 2019, respectively. His research interests include causal machine/deep learning, explainable AI, biomedical data analysis, and brain-computer interface. He serves as an editorial board member for Clinical and Molecular Hepatology (Artificial Intelligence Sector), Electronics, Frontiers in Neuroscience, Frontiers in Radiology (Artificial Intelligence in Radiology), International Journal of Imaging Systems and Technology (CIRSI), and a Program Committee or a Reviewer for NeurIPS, ICMIL, ICLR, AAAI, IJCAI, CVPR, MICCAI, AISTATS, etc.