# Learning Spatio-Temporal Manifold Representation for Probabilistic Land Deformation Prediction

Xovee Xu, Ting Zhong, Fan Zhou, Rongfan Li, Goce Trajcevski, and Qinggang Meng

***Abstract*—Landslides refer to occurrences of massive ground movements due to geological (and meteorological) factors, and can have disastrous impacts on property, economy, and even lead to the loss of life. The advances in remote sensing provide accurate and continuous terrain monitoring, enabling the study and analysis of land deformation which, in turn, can be used for land deformation prediction. Prior studies either rely on pre-defined factors and patterns or model static land observations without considering the subtle interactions between different point locations and the dynamic changes of the surface conditions, causing the prediction model to be less generalized and unable to capture the temporal deformation characteristics. To address these issues, we present DyLand, a dynamic manifold learning framework that models the dynamic structures of the terrain surface. We contribute to the land deformation prediction literature in four directions. First, DyLand learns the spatial connections of InSAR measurements and estimates the conditional distributions on a dynamic terrain manifold with a novel normalizing flow-based method. Second, instead of modeling the stable terrains, we incorporate surface permutations and capture the innate dynamics of the land surface while allowing for tractable likelihood estimations on the manifold. Third, we formulate the spatio-temporal learning of land deformations as a dynamic system and unify the learning of spatial embeddings and surface deformation. At last, extensive experiments on curated real-world InSAR datasets (land slopes prone to landslides) show that DyLand outperforms existing benchmark models.**

***Index Terms*—Land deformation prediction, geoscience, graph neural network, landslide, manifold learning.**

Manuscript received 17 November 2022; revised 27 February 2023 and 8 May 2023; accepted 17 June 2023. Date of publication 24 July 2023. This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 62072077 and 62176043, in part by the Natural Science Foundation of Sichuan Province, China, under Grant No. 2022NSFSC0505, and in part by the National Science Foundation under Grant SWIFT No. 2030249. *(Corresponding author: Fan Zhou.)*

Xovee Xu is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610054, Sichuan, China (e-mail: xovee@std.uestc.edu.cn).

Fan Zhou, and Rongfan Li are with the School of Inforamtion and Software Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan 610054, China (e-mail: fan.zhou@uestc.edu.cn; rongfanli1998@gmail.com).

Ting Zhong is with the University of Electronic Science and Technology of China, Chengdu, Sichuan, 610054, China, and also with the Kash Institute of Electronics and Information Industry, Kashi, Xinjiang, 84400, China (e-mail: zhongting@uestc.edu.cn).

Goce Trajcevski is with the Department of Electrical and Computer Engineering, Iowa State University, Ames, IA 50011, USA (e-mail: gocet25@@iastate.edu).

Qinggang Meng is with the Department of Computer Science, Loughborough University, Loughborough, LE11 3TU, U.K. (e-mail: q.meng@lboro.ac.uk).

Color versions of one or more of the figures in this paper are available at https://doi.org/10.1109/TCYB.2023.3291049.

Digital Object Identifier 10.1109/TCYB.2023.3291049



## I. Introduction

LANDSLIDES are among the most common catastrophic hazards in various areas, posing significant threats to human lives, properties, and vital infrastructures [1]. The causes of these hazards may come from a variety of complicated reasons, including natural and human activities, e.g., volcanic eruption, overuse of groundwater, cloudburst, earthquake, or even vegetation changes due to seasonal climate oscillations and global warming [2]. These factors often occur suddenly and are hard to predict, making it difficult and challenging to assess and manage landslide risks. A recent major landslide occurred at Hpakant, Myanmar, on July 2, 2020 (directly due to heavy rains), causing a death toll of at least 168 people [3]. Monitoring landslide-prone areas in real-time are important for early warning systems and landslide activity analytics [4]–[6], and the advanced geodetic techniques in remote sensing, such as Interferometric Synthetic Aperture Radar (InSAR) and Global Navigation Satellite System (GNSS), enable accurate monitoring of slope deformations and forecasting potential landslides, which have recently been studied [7]–[9].

Earlier works on landslide risk assessment and detection mainly depend on (1) multidisciplinary expert knowledge from geotechnics, meteorology, and sociopolitical sciences; (2) pre-defined influential factors such as climate, rock stability and slope gradient [10], [11]. These knowledge are used to estimate the probability of landsliding, design decisive features for landslide prediction, and provide us with explainable predictions about why and what factors contribute most to the occurrence of hazards. For example in [12], the authors conducted feature selection and then used $k$-means to identify the threshold of slope instability. Patterns discovered from remote sensing images provide vital information for landslide classification and subsequent decision-making. However, designing such systems requires extensive expert knowledge and effort and is usually limited to *post hoc* explanations. The studied areas and collected data are also relatively small-scale due to the constraints of labor resources and field surveys, narrowing the generalizability of the underlying system in places with different land environments and weather conditions. On the other side, data-driven and feature-engineering approaches are also restricted by the noises and uncertainties intrinsically

existed in the land deformation data, or failed to reflect the landslide predisposition due to data and metric biases [13].

Many studies [14], [15] learned landslide dynamics with machine learning techniques through leveraging the rich measurements of weather, surface, vegetation, and geology. Various traditional models have been exploited in the literature, including analytical hierarchy process [16], Bayesian networks [17], logistic regression [18], ensemble learning [19], and statistical learning [20]. However, they are still inadequate for probabilistic modeling and inference on the surface, which is essential for land stability estimation (inferring the distribution of deformations and understanding the mechanisms behind the deformations). While some recent works [21]–[23] estimate the distribution on the low-dimensional manifold, they focus on static surfaces. Estimating the density on a dynamic manifold remains a challenge.

More recently, the advances of deep neural networks have inspired several novel landslide prediction and inference models. Specifically, Convolutional Neural Networks (CNNs) are widely used for landslide susceptibility mapping [8], [24], which generally rely on inventory mapping from the InSAR data to outline the landslide boundaries and deformation features. For example, LACNN [8] is a locally aligned CNN model that takes the orientation of each pixel at multiple scales to capture hidden features, and assess the landslide susceptibility at a specific point. Graph Neural Networks (GNNs) are also used to model land deformations, e.g., SA-GNN [25] embeds the 3D surface into a 2D graph with locally linear embedding [26] and preserves the relative positions and slopes of adjacent points on the surface. It then uses a Spatial-Temporal GNN to capture the characteristics of neighboring points and predict the terrain deformations.

Despite the achieved promising results, existing land deformation models still confront several challenges. First, they either exploit 2D CNNs to extract the feature maps [8] or project the surface into a plain graph [25] – inevitably introducing errors and, more importantly, ignoring vital information (e.g., azimuth, orientations, slopes) for assessing the stability of the surface. Second, sequential methods which study the periodic displacements and influential factors ignore the spatial correlations of land locations [27]. Third, the InSAR observations, represented as point cloud, are associated with temporal deformations beyond geographical position of the monitored areas. Therefore, directly applying these methods to the point cloud manifold [28]–[30] are insufficient to capture the dynamic deformation patterns of the surface.

In this paper, we propose a novel probabilistic manifold learning model called DyLand for dynamic land deformation forecasting. Different from previous studies that project the 3D point clouds to 2D space, we directly model the spatial structures on the manifold without the loss of important geometric information caused by dimension reduction. We extend our previous work [30] by simultaneously capturing the topological dependencies and temporal deformations. DyLand considers the dynamics of surface and collective deformation trend in a unison, rather than learning them separately. As a principled dynamic manifold learning framework, DyLand generalizes normalizing flows [31] for density estimation on the manifold surface while preserving the intricate dynamics of the continuously changed terrains.

In sum, our main contributions are four-fold:

- We explicitly explore the land surface dynamics and topological dependencies between monitored locations. We illustrate the conceptual limitations of existing methods and tackle two fundamental drawbacks by learning dynamic manifold and unifying the co-evolution of temporal surface deformations and spatial representations.
- We propose to estimate the dynamic manifold density that has not been studied before. Our method generalizes flow-based generative models to learn a probability density over the manifold, while preserving the dynamics via a local deformation perturbation strategy.
- We formulate the spatio-temporal learning of land deformation as a dynamic system, and introduce neural ordinary differentiable equations to unify the learning of spatial embedding and surface deformation. We model the terrain dynamics in the continuous domain while better approximating the optimal deformation posterior.
- We collect real-world InSAR point cloud data of slope evolution and conduct extensive experiments to evaluate the effectiveness of the proposed model.

The reminder of this paper is organized as follows. In the next section, we review and discuss the related work. Section III introduces necessary preliminaries. We then present the details of DyLand methodology in Section IV, where we show how the spatial manifold dependencies and temporal land deformations are unified by dynamic manifold learning and probabilistic density estimation, respectively. Section V shows the performance of DyLand and baselines on real-world land deformation data and gives intuitive explanations and visualizations. At last, we conclude this paper in Section VI.

## II. Related Work

Previous work on landslide forecasting primarily focus on expert and statistical systems, manifold learning on the remote sensing maps, as well as convolutional neural networks and graph neural networks. We now review the literature that closely related to our work.

### A. Geometric and Manifold Learning

Geometric graph data are often represented as nodes, edges, and their associated features. They are arranged irregularly and may scale from tens to millions of nodes and edges, which brings difficulties in learning their effective representations and finding helpful data patterns. On the other side, manifold data refer to geometric shapes and surfaces (e.g., the terrain surface of the studied area). One goal of geometric and manifold learning is to describe these topological data in low dimensional space [32], which gives us a great convenience to adopt typical operations such as convolution and pooling to feasibly processing and analyzing them [33].

However, for a specific point in the manifold, it is not straightforward to find its nearest neighbors, nor to determine the number of them. In addition, judging point similarities in the original space and maintaining their properties in the

latent space are also critical for manifold learning algorithms. Existing works addressing these challenges in different ways. Locally linear embedding methods [34] represent each data point in the manifold using its nearest neighbors. In [35], the authors propose to measure the distance between two data points by their geodesic or shortest path. Student $t$-distribution is utilized in [36] to compute the similarity between two points in the latent space. Based on the assumption about data on Riemannian manifold, fuzzy topological structures are learned in [37]. More recent studies [38], [39] have shown that flow-based generative methods can learn better manifold structures by adopting invertible and bijective mappings, achieving state-of-the-art performance. Other aspects of manifold learning, e.g., denoising and out-of-distribution detection [40], can be implemented simultaneously. Different from existing static manifold methods, our approach integrates dynamic manifold by explicitly considering the temporal land deformations.

### B. Land Deformation Forecasting and Susceptibility Mapping

Forecasting the deformation of landslides is a long-standing research topic that aims to monitor the terrain surface and assess the risk of catastrophic hazards such as landslides, land subsidence, debris and mudflows which could pose significant threats to human lives, properties, and vital infrastructures [1]. They occur due to the gravity and/or elevation, and deformations of the terrain surface caused by numerous factors, such as water infiltration, glacier melting, aquifer exploitation, rock erosion, earthquakes, and volcanic eruptions [2]. Traditional landslide learning and prediction methods largely rely on handcrafted features and expert domain knowledge. For example, the technology of geographical information system (GIS) and probabilistic modeling are combined to produce susceptibility maps based on multi-layer spatial data [41]. The satellite remote sensing images are incorporated into an expert decision support system to conduct image classifications and multi-category threshold identification based on slope instability [12]. However, these methods often face the problems of data sparsity and poor generalization capability.

High precision InSAR and remote sensing techniques are now ubiquitously used by researchers and practitioners to monitor land surface deformation and identify unstable land slopes. In addition to common features like texture and color extracted from images, many works resort to automatic detection/learning methods. In particular, the advances of deep neural networks and rapidly increased computing power have inspired many novel methods for landslide-related tasks in recent years [24], [25]. Among them, CNNs have been established as a standard tool for learning visual knowledge from aerial images. For example, [42] adopts CNNs to recognize landslide based on texture changes. CNNs and GANs are utilized to obtain high-quality InSAR images via image super-resolution techniques [43]–[45]. Another example is LACNN [8], the authors create a two-stage CNN framework to generate the desired probability map: a down-sampling stage for capturing the local and global properties of the input image; and an up-sampling stage based on interpolation. LACNN incorporates orientational information into CNNs which is suitable for landslide susceptibility mapping. One notable limitation of CNN-based methods is the lack of modeling spatial correlations in non-Euclidean space.

Based on the InSAR point cloud maps that contain rich spatial information, graph-based approaches are proposed for land deformation forecasting. Point cloud data can be naturally represented as nodes and edges in GNNs. Both internal and external factors such as weather, point distance and soil condition can be used as node/edge features for data aggregation in non-Euclidean space. In [25], the authors proposed SA-GNN model for landslide prediction. Specifically, SA-GNN designs a slope-aware locally linear embedding module for preserving the spatial characteristics of a point cloud. Despite their improved performance in forecasting land deformations, temporal dynamics of the land surface are often absent. In this work, we unify the spatial and temporal learning processes into a dynamic system and introduce a novel normalizing flow-based method for estimating the density of manifold surfaces in the continuous-time domain. The proposed method relieves the need for feature engineering and is capable of simultaneously incorporating various landslide-related factors. Unlike CNN-based models, which overlook vital surface information such as azimuth and orientation, our model utilizes InSAR point cloud data to accurately represent the monitored locations. Moreover, we extend static point cloud representations to encompass topological dependencies and temporal deformations, unifying them for enhanced land deformation prediction.

TABLE I
MATHEMATICAL NOTATIONS

| Symbol | Description |
|---|---|
| $\mathbf{A}$ | Adjacency matrix of the point cloud $\mathbf{V}$. |
| $M$ | Dimension of the latent variables. |
| $N$ | Number of monitored locations in the studied area. |
| $\mathbf{S}$ | Temporal land deformation observations. |
| $T$ | Time. |
| $\mathbf{V}$ | Point cloud of terrain surface, consisting of $N$ locations. |
| $\mathbf{W}$ | Spatial representation of the point cloud $\mathbf{V}$. |
| $\mathbf{U}$ | Latent factor encodes the point cloud locations $\mathbf{V}$ and temporal deformation $\mathbf{S}$. |
| $y_i^\tau$ | The deformation of the $i$-th location at time $\tau$. |
| $\mathbf{Y}$ | Predicted deformations. |

## III. PRELIMINARIES

In this section, we first formally define the land deformation forecasting problem, and then give necessary background about spatio-temporal graph neural networks (ST-GNNs). The frequently used mathematical notations are summarized in Table I.

The InSAR point cloud data corresponds to a collection of $N$ monitored locations, each with a unique coordinate (longitude, latitude, and elevation). Then the point cloud can be represented as a vector of triplets $\mathbf{V} \in \mathbb{R}^{N\times 3}$. Each location is associated with a temporal sequence which is the deformation observations, denoted as $\mathbf{S} \in \mathbb{R}^{N\times T}$. Here $T$ is the sequence length. The land deformations of the point cloud at timestamp $\tau$ are denoted as $\mathbf{S}^\tau \in \mathbb{R}^{N\times 1}$. Then we formulate the *land deformation forecasting* problem as follows:

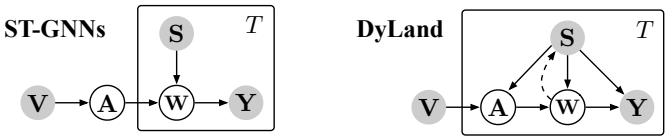



Fig. 1. Illustration of the difference between existing ST-GNNs and ours. Point cloud locations $\mathbf{V}$ and temporal land deformations $\mathbf{S}$ are inputs of the model, $\mathbf{Y}$ is the predicted deformations. Adjacency $\mathbf{A}$ and spatio-temporal representation $\mathbf{W}$ are derived from $\mathbf{V}$ and $\mathbf{S}$.

*Definition 1 (**Land Deformation Forecasting**):* For a studied area with $N$ spatial locations $\mathbf{V}$ and their associated temporal deformation observations $\mathbf{S}$, we aim to forecast the surface deformations $\mathbf{Y} \in \mathbb{R}^{N\times T'}$ of all $N$ locations in future timestamps $T'$.

This is a typical spatio-temporal prediction problem, given spatial locations $\mathbf{V}$ and temporal deformations $\mathbf{S}$, we predict the future surface deformation $\mathbf{Y}$:

$$p(\mathbf{Y}|\mathbf{V},\mathbf{S}). \tag{1}$$

To explicitly model the spatio-temporal dependencies between terrain surface point locations, GNNs are utilized as learning backbones in existing works [46]. Specifically, for cloud points $\mathbf{V}$, an adjacency matrix $\mathbf{A}$ is introduced to learn their spatio representations $\mathbf{W}$. Here we need an alternative implementation of Eq. (1) according to Bayes' rule:

$$p(\mathbf{Y},\mathbf{W},\mathbf{A}|\mathbf{V},\mathbf{S}) = p(\mathbf{A}|\mathbf{V},\mathbf{S})p(\mathbf{Y},\mathbf{W}|\mathbf{A},\mathbf{V},\mathbf{S}) \tag{2}$$

$$= \underbrace{p(\mathbf{A}|\mathbf{V},\mathbf{S})}_{\textbf{Adjacency}}\underbrace{p(\mathbf{W}|\mathbf{A},\mathbf{V},\mathbf{S})}_{\textbf{GNN learning}}\underbrace{p(\mathbf{Y}|\mathbf{W},\mathbf{A},\mathbf{V},\mathbf{S})}_{\textbf{Prediction}}, \tag{3}$$

where the first term $p(\mathbf{A}|\mathbf{V},\mathbf{S})$ denotes the adjacency matrix construction; the second term $p(\mathbf{W}|\mathbf{A},\mathbf{V},\mathbf{S})$ denotes the spatio-temporal learning of data structures and historical observations, which is generally implemented by a specific GNN architecture, e.g., GCN or GAT; and the last term is the future temporal forecasting desired by downstream tasks (e.g., traffic flows [47] or land deformations studied in this work) that can be realized by any time-series learning methods, such as RNNs and ARIMA [48].

## IV. Methodology: DyLand

In this section, we propose a probabilistic manifold learning model (DyLand) for land deformation forecasting. We first state our motivation and give an overview of DyLand. Next, we detail the dynamic manifold learning and spatial representation extraction. Then, a unified module based on neural ordinary differential equations (ODEs) is proposed for deformation forecastings. Finally, the optimization and training processes are illustrated.

### A. Motivation and Overview

In existing spatio-temporal GNNs, the adjacency matrix $\mathbf{A}$ is often built by $p(\mathbf{A}|\mathbf{V})$ as the positions of point cloud $\mathbf{V}$ are generally fixed, e.g., the regions in flow prediction [48] and sensors in traffic forecasting [49], [50]. This, however, may not fully capture the complex interactions between different point locations since the land surfaces are continuously deformed.

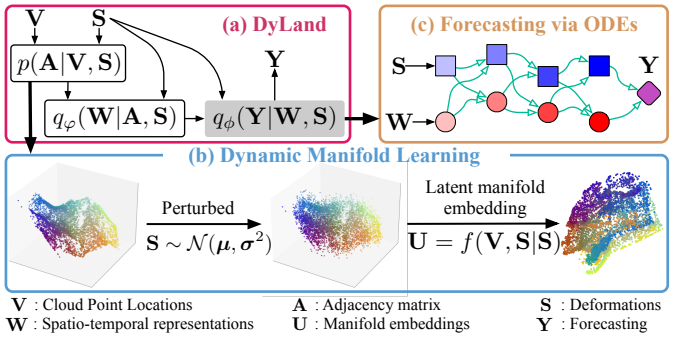



Fig. 2. **(a)** Overview of DyLand. **(b)** We learn $p(\mathbf{A}|\mathbf{V},\mathbf{S})$ through mapping both temporal deformation $\mathbf{S}$ and spatial locations $\mathbf{V}$ in a dynamic manifold space. **(c)** We forecast and extrapolate the surface deformations $p(\mathbf{Y}|\mathbf{W},\mathbf{S})$ via neural ODEs.

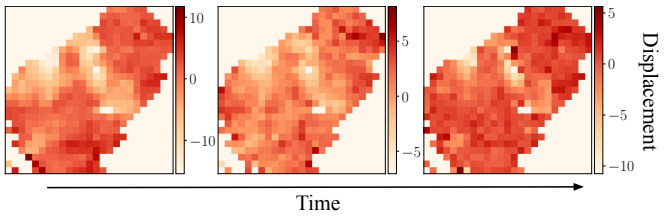



Fig. 3. Dynamic land displacement heat maps (real terrain).

The spatio-temporal learning of landslides requires a joint distribution $p(\mathbf{Y},\mathbf{W}|\mathbf{A},\mathbf{V},\mathbf{S})$ according to Eq. (2). However, existing methods often learn spatial representation $\mathbf{W}$ directly from $\boldsymbol{\Omega} = \{\mathbf{A},\mathbf{V},\mathbf{S}\}$ which is conditionally independent of $\mathbf{Y}$, i.e., $\boldsymbol{\Omega} \perp\!\!\!\perp \mathbf{Y}|\mathbf{W}$. Thus, the spatio-temporal joint distribution is decomposed into two steps:

$$p(\mathbf{Y},\mathbf{W}|\boldsymbol{\Omega}) = p(\mathbf{W}|\boldsymbol{\Omega})p(\mathbf{Y}|\mathbf{W}), \tag{4}$$

where the deformation forecasting $p(\mathbf{Y}|\mathbf{W})$ is made without considering the dynamics of cloud points. We illustrate the difference between existing ST-GNNs and DyLand in Fig. 1.

An overview of DyLand is given in Fig. 2. Specifically, we tackle two fundamental drawbacks of previous methods.

- First, we emphasize the importance of temporal deformation information $\mathbf{S}$ beyond Euclidean positions in learning $p(\mathbf{A}|\mathbf{V},\mathbf{S})$ through mapping both $\mathbf{S}$ and spatial locations $\mathbf{V}$ in the manifold space, where the similarity among cloud points and deformations can be better measured. Here, we propose a probabilistic generative model to learn $p(\mathbf{A}|\mathbf{V},\mathbf{S})$ by incorporating the dynamics of land surface (cf. Fig. 2 (b) and Section IV-B).
- Second, we model the conditional distribution $p(\mathbf{Y}|\mathbf{W},\mathbf{S})$ jointly with $\mathbf{W}$ and $\mathbf{S}$ rather than solely relying on $\mathbf{W}$ as in previous methods (cf. Fig. 1). In this way, given spatial cloud points $\mathbf{V}$ and $\mathbf{A}$ modeled by $p(\mathbf{A}|\mathbf{V},\mathbf{S})$, we are able to better approximate the optimal deformation forecasting. Toward this goal, we consider the surface deformations as a dynamic system and present an approach based on neural ODEs [51] to predict and extrapolate temporal deformations as $p(\mathbf{Y}|\mathbf{W},\mathbf{S})$ (cf. Fig. 2 (c)) and Sections IV-C and IV-D.

### B. Dynamic Manifold Learning

Land displacement is inherently a dynamic process. As shown in Fig. 3, for a specific position in the studied area, its displacement generally follows the trends of neighboring positions over time. When using GNNs to model such trends [25], the continuous displacement of the land surface is ignored because of the static nature of adjacency matrix $\mathbf{A}$. Besides, the influence of points with the same distance may not be consistent since the relative positions (e.g., slope and azimuth) of neighboring points are of great importance for informing the displacement trends [8], [25].

To train a model that is able to estimate the land deformation distribution and adapt the static adjacency $\mathbf{A}$ to dynamic, we introduce a latent factor $\mathbf{U}$ and assume a deterministic mapping $p(\mathbf{A}|\mathbf{U})$ as:

$$p(\mathbf{A}|\mathbf{V},\mathbf{S}) = \sum_{\mathbf{U}} p(\mathbf{A}|\mathbf{U})p(\mathbf{U}|\mathbf{V},\mathbf{S}), \tag{5}$$

where the dynamic adjacency generation considers both geographical locations $\mathbf{V}$ and temporal deformation evolution $\mathbf{S}$.

The latent factor $\mathbf{U}^\tau$ at a specific timestamp $\tau \in [1,T]$ is estimated by a stochastic model $\mathbf{U}^\tau = f_{\mathbf{U}}(\mathbf{V}^\tau, \mathbf{S}^\tau|\mathbf{S}^\tau)$, which jointly generalizes the geographical similarity and local deformation trend. For the $i$-th cloud point, the latent factor $\mathbf{u}_i \in \mathbf{U}$ encodes the Euclidean spatial dependency and temporal deformations of adjacent areas.

Since the land surface is a typical manifold (i.e., 2D manifold data scattered over 3D space) and is continuously displaced, we need to estimate the distribution of cloud points in non-Euclidean space in general. More specifically, we need a density estimation approach to infer the surface manifold directly rather than learning it in the 2D space which would inevitably suffer information loss. A few efforts have been made to probability density estimation over the manifold. For example, [21] first defines normalizing flows (NFs) and estimates the probability density on a Riemannian manifold embedded in high-dimensional data space. In [22], [23], the authors investigate the expressiveness and stability of tractable density on the space that are not diffeomorphic to Euclidean such as tori and sphere. However, these works define NFs on static manifold, which cannot be applied to dynamic surface.

To address the aforementioned obstacles, we present an NF-based approach to capture the cloud point interactions and deformation dynamics. Specifically, we add the deformation $\mathbf{s}_i^\tau$ to the $i$-th monitored location $\mathbf{v}_i$ and estimate the distribution of perturbed vector as $\mathbf{v}_i' = f^{-1}(\mathbf{u}_i^\tau|\mathbf{s}_i^\tau)$. Then we learn a transformation from latent factor $\mathbf{u}_i^\tau$ to perturbed location $\mathbf{v}_i'$:

$$\log p(\mathbf{v}_i'|\mathbf{s}_i^\tau) = \log p(\mathbf{u}_i^\tau) - \log \left| \det \frac{\partial f^{-1}(\mathbf{u}_i^\tau|\mathbf{s}_i^\tau)}{\partial \mathbf{u}_i^\tau} \right|. \tag{6}$$

Here we use invertible mappings to transform the uniform distribution $p(\mathbf{u}_i^\tau)$ to the desired complex conditional distribution $p(\mathbf{v}_i'|\mathbf{s}_i^\tau)$. Among various NF variants (cf. [31] for a comprehensive review), we choose the continuous NF [52], [53] as it imposes no restrictions on model architecture and the log-density follows the instantaneous change of variable formula, which can be described as:

$$\mathbf{u}^{t_i} = \mathbf{u}^{t_0} + \int_{t_0}^{t_1} f^{-1}(\mathbf{u}^t, t, \mathbf{s}_i^\tau)dt, \tag{7}$$

$$\log p(\mathbf{u}^{t_1}|\mathbf{s}_i^\tau) = \log p(\mathbf{u}^{t_0}) - \int_{t_0}^{t_1} \mathrm{Tr}\left(\frac{\partial f^{-1}}{\partial \mathbf{u}^t}\right) dt, \tag{8}$$

where $\mathbf{u}^{t_1} = \mathbf{v}_i'$ and $\mathbf{u}^{t_0} = \mathbf{u}_i^\tau$, and the trace can be efficiently computed by Hutchinson's estimator [54]. The dimensions of latent factors are restricted to be exactly the same by Eqs. (6) and (8), which limit the information density of representations in high dimensional space. Recent studies suggest to use fully-connected (FC) layers to bridge the gap between different dimension scales, however, such transformations are difficult when the dimensions of the original space are very high, but in our case, the transformation of 3D surface of $\mathbf{U}$ is feasible and easy to train.

Above we have exploited NFs to capture spatial correlations of land surface and dynamics of land deformation, i.e., $p(\mathbf{U}|\mathbf{V},\mathbf{S})$. Then, for any two points with learned latent factors $\mathbf{u}_i$ and $\mathbf{u}_j$, the mapping $p(\mathbf{A}|\mathbf{U})$ is defined as:

$$p(\mathbf{A}^\tau|\mathbf{U}^\tau) = \prod_{i=1}^{N}\prod_{j=1}^{N} p(A_{ij}^\tau|\mathbf{u}_i^\tau, \mathbf{u}_j^\tau) \tag{9}$$

$$\text{with} \quad p(A_{ij}^\tau = 0|\mathbf{u}_i^\tau, \mathbf{u}_j^\tau) = \mathrm{sigmoid}(\|\mathbf{u}_i^\tau - \mathbf{u}_j^\tau\|_2), \tag{10}$$

where $A_{ij}^\tau \in \mathbf{A}^\tau$ and the similarity between the two points in the latent space is estimated with a logistic sigmoid function.

In Algorithm 1, we summarize the training process of DyLand's dynamic manifold learning. The time complexity for computing the unbiased stochastic estimator of the likelihood is $\mathcal{O}(d)$, where $d$ is the dimension of $\mathcal{V}$. The complexity for evaluating $f$ is $\mathcal{O}(dC)$, where $C$ is the largest size of hidden layers. Overall, the complexity for computing the likelihood is $\mathcal{O}(dL(C+1))$, here $L$ is the number of evaluations of $f$ used by the ODE solver [53].

**Algorithm 1** Dynamic Manifold Learning

**Input:** Point cloud locations $\mathbf{V} \in \mathbb{R}^{N\times d}$, land deformations $\mathbf{S} \in \mathbb{R}^{N\times T}$, latent factor $\mathbf{U}$, manifold mapping $f$, start time $t_0$, stop time $t_1$, Hutchinson's estimator $\mathbb{E}_H$, and pre-determined ODE solver.

**function** $f_{aug}^{-1}([\mathbf{u}^t, \mathbf{s}], t)$: ▷ Define augment $f$ with trace estimation
  $\mathrm{Tr} = -\mathbb{E}_H(f^{-1}, \mathbf{u}^t)$; ▷ Trace via Hutchinson's estimator $\mathbb{E}_H$
  $f_t = f^{-1}(\mathbf{u}^t, \mathbf{s}, t)$; ▷ Transform the input by $f^{-1}$ via Eq. (7)
  **return** $[f_t, \mathrm{Tr}]$;
**end function**
Initialize $\tau = 1, i = 1, \mathcal{L}_{\mathbf{U}} = 0$;
**while** $\tau \le T$ **do** ▷ For each time in the sequence
  **while** $i \le N$ **do** ▷ For each location in the point cloud
    $\mathbf{u}^{t_1} = \mathbf{v}_i + \mathbf{s}_i^\tau$; ▷ Perturb the monitored locations
    $[\mathbf{u}^{t_0}, \mathrm{Tr}] = \mathrm{Solver}(f_{aug}^{-1}, [\mathbf{u}^{t_1}, \mathbf{s}_i^\tau], t_0, t_1)$; ▷ Solve the ODE
    $\log \hat{p}(\mathbf{u}^{t_1}|\mathbf{s}_i^\tau) = \log p(\mathbf{u}^{t_0}) - \mathrm{Tr}$ ; ▷ Estimate log density
    $\mathcal{L}_{\mathbf{U}} = \mathcal{L}_{\mathbf{U}} - \log \hat{p}(\mathbf{u}^{t_1}|\mathbf{s}_i^\tau)$; ▷ Add up loss
    $\mathbf{u}_i^\tau = \mathbf{u}^{t_0}$; ▷ Get the dynamic manifold embeddings
  **end while**
**end while**
Minimize $\mathcal{L}_{\mathbf{U}}$ and obtain the latent factor $\mathbf{U}$ via Eq. (8);

**Algorithm 2** Training process of estimating $p(\mathbf{W}|\mathbf{A},\mathbf{V},\mathbf{S})$

**Input:** Dynamic adjacency $\mathbf{A}^\tau$, land deformations $\mathbf{S}^\tau$.
**Output:** Spatial representation $\mathbf{W}^\tau$.
1: $\boldsymbol{\mu}_{\mathbf{W}} = \text{GNN}_{\boldsymbol{\mu}_{\mathbf{W}}}(\mathbf{A},\mathbf{S})$; ▷ Get mean via VGAE
2: $\log \boldsymbol{\sigma}_{\mathbf{W}} = \text{GNN}_{\boldsymbol{\sigma}_{\mathbf{W}}}(\mathbf{A},\mathbf{S})$; ▷ Get log standard deviation via VGAE
3: Sample $\boldsymbol{\epsilon}$ from standard Gaussian $\mathcal{N}(\mathbf{0},\mathbf{I}^2)$;
4: $\mathbf{W} = \boldsymbol{\mu}_{\mathbf{W}} + \boldsymbol{\epsilon}\boldsymbol{\sigma}_{\mathbf{W}}$; ▷ Reparameterization trick via Eq. (14)
5: $\widehat{\mathbf{S}} = \text{FC}(\mathbf{W})$; ▷ Reconstruct $\widehat{\mathbf{S}}$ via Eq. (15)
6: Compute encoder loss by $\boldsymbol{\mu}_{\mathbf{W}}$ and $\boldsymbol{\sigma}_{\mathbf{W}}$ (cf. Eq. (13)), then add it to the reconstruction loss to get the ELBO in Eq. (12);

### C. Extracting Spatial Representations

To approximate $p(\mathbf{W}|\mathbf{\Omega})$ by $q_\varphi(\mathbf{W}|\mathbf{\Omega})$, we need to maximize the evidence lower bound (ELBO):

$$\log p(\mathbf{A},\mathbf{S}) - D_{\text{KL}}(q_\phi(\mathbf{W}|\mathbf{A},\mathbf{S}) \parallel p(\mathbf{W}|\mathbf{A},\mathbf{S})) \tag{11}$$

$$= \mathbb{E}_{q_\varphi}[\log p(\mathbf{A},\mathbf{S}|\mathbf{W})] - D_{\text{KL}}(q_\varphi(\mathbf{W}|\mathbf{A},\mathbf{S}) \parallel p(\mathbf{W})), \tag{12}$$

where $\mathbf{V} \perp\!\!\!\perp \mathbf{W}|\mathbf{A}$ and $p(\mathbf{W}|\mathbf{\Omega}) = p(\mathbf{W}|\mathbf{A},\mathbf{S})$. Instead of approximating standard Gaussian $\mathcal{N}(\mathbf{0},\mathbf{I}^2)$ conventionally, we argue that, $\mathbf{W}$ should maintain a similar distribution to $\mathbf{S}$, where it is aggregated from:

$$\begin{aligned} & D_{\text{KL}}\left(q_\varphi\left(\mathbf{W}^\tau|\mathbf{A}^\tau,\mathbf{S}^\tau\right) \parallel p(\mathbf{W}^\tau)\right) \\ \simeq & D_{\text{KL}}\left(q_\varphi(\mathbf{W}^\tau|\mathbf{A}^\tau,\mathbf{S}^\tau) \parallel \mathcal{N}\left(\boldsymbol{\mu}^\tau, {\boldsymbol{\sigma}^\tau}^2\right)\right) \\ = & D_{\text{KL}}\left(\mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{w}}, \sigma^2_{\mathbf{W}}\right) \parallel \mathcal{N}\left(\boldsymbol{\mu}^\tau, {\boldsymbol{\sigma}^\tau}^2\right)\right) \\ = & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}\left(2\log\frac{\sigma_i^j}{{\sigma_{\mathbf{w}}}_i^j} + \frac{\left(\mu_i^j - {\mu_{\mathbf{w}}}_i^j\right)^2}{\left(\sigma_i^j\right)^2} + \frac{\left({\sigma_{\mathbf{w}}}_i^j\right)^2}{\left(\sigma_i^j\right)^2} - 1\right), \end{aligned} \tag{13}$$

where $M$ is the dimension of latent variables, and $\mathbf{W}^\tau \in \mathbb{R}^{N\times M}$. In practice, $\mathbf{W}^\tau$ can be extracted from dynamic adjacency $\mathbf{A}^\tau$ and $\mathbf{S}^\tau$ by a reparameterization trick as follows:

$$\mathbf{W}^\tau = \text{GNN}_{\boldsymbol{\mu}_{\mathbf{W}}}(\mathbf{A}^\tau,\mathbf{S}^\tau) + \boldsymbol{\epsilon}\exp(\text{GNN}_{\boldsymbol{\sigma}_{\mathbf{W}}}(\mathbf{A}^\tau,\mathbf{S}^\tau)), \tag{14}$$

where $\mathbf{W}^\tau \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{W}}, \boldsymbol{\sigma}^2_{\mathbf{W}})$ and $\boldsymbol{\epsilon}$ is sampled from standard Gaussian. We implement the GNN by VGAE [55]. The reconstruction term (a.k.a. decoder) in Eq. (12) is estimated by Monte Carlo:

$$\mathbb{E}_{q_\varphi}\left[\log p(\mathbf{A}^\tau,\mathbf{s}^\tau|\mathbf{W}^\tau)\right] \simeq \frac{1}{N}\sum_{i=1}^{N}\log p_\xi(\hat{s}_i^\tau|\mathbf{w}_i^\tau). \tag{15}$$

For simplicity, $p_\xi$ is FC layers. Other network architectures can be used here for performance improvement. Algorithm 2 summarizes the training process for estimating the conditional distribution $p(\mathbf{W}|\mathbf{A},\mathbf{V},\mathbf{S})$. The time complexity for computing the 2-layer graph convolution is $\mathcal{O}(2EM+2NM^2)$, which is linear to the number of nodes $N$ and number of edges $E$.

### D. Unifying Deformation and Representation Learning

Now we have obtained the spatial representations $\mathbf{W}$. This section shows how to unify $\mathbf{W}$ with temporal deformations $\mathbf{S}$ for final forecastings.

Recall that $\mathbf{W}$ is learned by GNNs – which often suffer from over-smoothing problem due to the underlying aggregation mechanism, especially when they are going deeper [46]. Although [56] suggests adding an extra normalization layer to prevent the representations from being too similar, it may disturb the regular feature aggregation essential for learning the local deformation trends and thus deteriorate the forecasting performance. Meanwhile, existing ST-GNNs [25], [48] make temporal forecastings based on $p(\mathbf{Y}|\mathbf{W})$ which implicitly assuming $\mathbf{\Omega} \perp\!\!\!\perp \mathbf{Y}|\mathbf{W}$ (cf. Eq. (4)). However, we argue that $\mathbf{S}$ is indispensable for making better forecastings. Therefore, we propose to model the co-evolution of temporal land deformations and spatial representations as a dynamic system.

Specifically, we build a function $g$ parameterized by neural networks to model the hidden dynamics of $\mathbf{S}$ and $\mathbf{W}$ from current time $t_0$ to forecasting time $t_1$. The function $g$ is a homeomorphism, which ensures the predicted deformations preserve the topology of the terrain surface. The forecasting process is defined as:

$$\begin{bmatrix}\mathbf{S}^{t_1}\\ \mathbf{W}^{t_1}\end{bmatrix} = \begin{bmatrix}\mathbf{S}^{t_0}\\ \mathbf{W}^{t_0}\end{bmatrix} + \int_{t_0}^{t_1} g\left(\begin{bmatrix}\mathbf{S}^{t}\\ \mathbf{W}^{t}\end{bmatrix}, t\right) dt, \tag{16}$$

where $t_0 \in [1,T], t_1 \in [T+1,T']$, $\mathbf{S}^{t_1}$ is exactly the forecasting $\widetilde{\mathbf{Y}}$ at time $t_1$. The ODEs are solved with adaptive Runge-Kutta 4(5) scheme of Dormand-Prince [57]. Let $\mathbf{z}_0 = [\mathbf{S}^{t_0}, \mathbf{W}^{t_0}]^\intercal$, at step $n+1$, the $p$-order approximation of $\mathbf{z}_{n+1}$ is given by:

$$\mathbf{z}_{n+1}^p = \mathbf{z}_n^p + h\sum_{i=1}^{l} b_i^p \mathbf{k}_i, \quad n = 0,1,2,\ldots, \tag{17}$$

$$\text{with} \quad \mathbf{k}_i = g(\mathbf{z}_n^p + h\sum_{j=1}^{i-1} a_{i,j}\mathbf{k}_j, t_n + c_i h), \tag{18}$$

where $h$ is an adaptive step size; vector $\mathbf{k}$ denotes the slopes of $g$; parameters $l$, $a$, $b$ and $c$ are all arranged in a mnemonic device – a.k.a. Butcher tableau. The local truncation error is on the order of $O(h^5)$, and is calculated as:

$$\mathbf{e}_{n+1} = \mathbf{z}_{n+1}^t - \mathbf{z}_{n+1}^4 = h\sum_{i=1}^{l}\left(b_i^5 - b_i^4\right)\mathbf{k}_i. \tag{19}$$

This error is used to tune $h$ by giving a maximum error $\mathbf{e}_{\text{max}}$: we half the step size when $\mathbf{e}_{n+1} \geq \mathbf{e}_{\text{max}}$; otherwise, we double the step size. We can see that larger $\mathbf{e}_{\text{max}}$ expedites the approximation while smaller $\mathbf{e}_{\text{max}}$ slows it. Algorithm 3 summarizes the training process of $\mathbf{S}$ and $\mathbf{W}$, which complexity is almost the same as the one in Algorithm 1, i.e., $\mathcal{O}(dL(C+1))$.

### E. Overall Optimization

We now describe how we approximate the desired probability distributions and the overall optimization of DyLand.

*1) Approximating $p(\mathbf{Y},\mathbf{W}|\mathbf{\Omega})$:* Existing methods ignore the prior distributions of forecasting $\mathbf{Y}$ and spatio-temporal representation $\mathbf{W}$ when parameterizing $p(\mathbf{Y},\mathbf{W}|\mathbf{\Omega})$. Thus, we propose a variational framework to approximate the intractable $p(\mathbf{Y},\mathbf{W}|\mathbf{\Omega})$ by $q_\rho(\mathbf{Y},\mathbf{W}|\mathbf{\Omega})$. The Kullback–Leibler (KL) di-

**Algorithm 3** Training process of estimating $p(\mathbf{Y}|\mathbf{W}, \boldsymbol{\Omega})$ via co-evolution of the spatial-temporal representations.

**Input:** Temporal deformations $\mathbf{S} \in \mathbb{R}^{N\times T}$, spatial representations $\mathbf{W}$, ground truth $\mathbf{Y}$, Hutchinson's estimator $\mathbb{E}_H$, predetermined ODE solver.
**Output:** Deformation forecastings $\widehat{\mathbf{Y}}$

**function** $g_{aug}(\mathbf{z}^t, t)$: ▷ Define augment $g$ with trace estimation
  $\mathrm{Tr} = -\mathbb{E}_H(g, \mathbf{z}^t)$; ▷ Trace via Hutchinson's estimator $\mathbb{E}_H$
  $g_t = g(\mathbf{z}^t, t)$; ▷ Transform the input by $g$ via Eq. (16)
  **return** $[g_t, \mathrm{Tr}]$;
**end function**
Initialize $\tau = 1, i = 1, \iota = T + 1$;
**while** $\iota \leq T'$ **do** ▷ Iteratively make forecasting from $T+1$ to $T'$
  $t_1 = \iota$ ;
  **while** $\tau \leq T$ **do** ▷ For each time in the sequence
    $t_0 = \tau$;
    **while** $i \leq N$ **do** ▷ For each location in the point cloud
      $\mathbf{z}^{t_0} = [\mathbf{s}_i^\tau, \mathbf{w}_i^\tau]^\intercal$; ▷ Initialize the vector
      $[\mathbf{z}^{t_1}, \mathrm{Tr}] = \mathrm{Solver}(g_{\mathrm{aug}}, \mathbf{z}^{t_0}, t_1, t_0)$; ▷ ODE via Eq. (17)
      $\widetilde{\mathbf{Y}}_i^\tau = \mathbf{z}^{t_1}$; ▷ Obtain future deformations
    **end while**
  **end while**
**end while**
$\widehat{\mathbf{Y}} = \mathrm{FC}([\widetilde{\mathbf{Y}}]_i)$; ▷ Aggregate forecastings
Calculate loss $\mathcal{L}_{\mathbf{Y}} = \mathcal{L}_{\mathrm{MSE}}(\widehat{\mathbf{Y}}, \mathbf{Y}) + \mathcal{L}_{\mathrm{ELBO}}(\widehat{\mathbf{Y}})$ via Eq. (25);

vergence measures how one probability distribution is different from the other one [58], which is derived as:

$$
\begin{aligned}
&D_{\mathrm{KL}}(q_\rho(\mathbf{Y}, \mathbf{W}|\boldsymbol{\Omega}) \parallel p(\mathbf{Y}, \mathbf{W}|\boldsymbol{\Omega})) \qquad (20)\\
=&\mathbb{E}_{\mathbf{Y},\mathbf{W}\sim q_\rho}[\log \frac{q_\rho(\mathbf{Y}, \mathbf{W}|\boldsymbol{\Omega})}{p(\mathbf{Y}, \mathbf{W}|\boldsymbol{\Omega})}]\\
=&\mathbb{E}_{\mathbf{Y},\mathbf{W}\sim q_\rho}[\log \frac{q_\varphi(\mathbf{W}|\boldsymbol{\Omega}) q_\phi(\mathbf{Y}|\mathbf{W}, \boldsymbol{\Omega})}{p(\mathbf{W}|\boldsymbol{\Omega}) p(\mathbf{Y}|\mathbf{W}, \boldsymbol{\Omega})}]\\
=&\mathbb{E}_{\mathbf{W}\sim q_\varphi}[\log \frac{q_\varphi(\mathbf{W}|\boldsymbol{\Omega})}{p(\mathbf{W}|\boldsymbol{\Omega})}] + \mathbb{E}_{\mathbf{Y}\sim q_\phi}[\log \frac{q_\phi(\mathbf{Y}|\mathbf{W}, \boldsymbol{\Omega})}{p(\mathbf{Y}|\mathbf{W}, \boldsymbol{\Omega})}]\\
=&D_{\mathrm{KL}}(q_\varphi(\mathbf{W}|\boldsymbol{\Omega}) \parallel p(\mathbf{W}|\boldsymbol{\Omega})) \qquad (21)\\
&+ D_{\mathrm{KL}}(q_\phi(\mathbf{Y}|\mathbf{W}, \boldsymbol{\Omega}) \parallel p(\mathbf{Y}|\mathbf{W}, \boldsymbol{\Omega})), \qquad (22)
\end{aligned}
$$

where $\rho$, $\varphi$ and $\phi$ are parameters of $q(\mathbf{Y}, \mathbf{W}|\boldsymbol{\Omega})$, $q(\mathbf{W}|\boldsymbol{\Omega})$ and $q(\mathbf{Y}|\mathbf{W}, \boldsymbol{\Omega})$, respectively (denoted as $q_\rho$, $q_\varphi$ and $q_\phi$ for short). According to Eq. (20) we can approximate distribution $p(\mathbf{Y}, \mathbf{W}|\boldsymbol{\Omega})$ by the KL divergences of marginals.

*2) Approximating $p(\mathbf{Y}|\mathbf{W}, \boldsymbol{\Omega})$:* For observed historical land deformations $[(\mathbf{S}_i^\tau, \mathbf{W}_i^\tau)]_{\tau=0,1,\ldots,T}$ at each cloud point position $i$, the final land deformation forecastings $\widehat{\mathbf{Y}} \in \mathbb{R}^{N\times T'}$ are obtained by aggregating the $T$ temporal deformation representation $\widetilde{\mathbf{Y}} \in \mathbb{R}^{N\times T\times T'} = \left[\widetilde{\mathbf{Y}}_i^\tau\right]_{\tau=0,1,\ldots,T}$ via FC layers. Each of $\widetilde{\mathbf{Y}}_i^\tau$ is predicted by the ODE module as in Eq. (16). Besides, the deformations $\mathbf{Y}^\tau$ at a specific time should follow the prior distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ by approximating the optima $p(\mathbf{Y}^\tau|\mathbf{W}^\tau, \mathbf{S}^\tau)$ with a proposal distribution $q_\phi(\mathbf{Y}^\tau|\mathbf{W}^\tau, \mathbf{S}^\tau)$. This is achieved by minimizing the $D_{\mathrm{KL}}$ term in Eq. (21):

$$
\begin{aligned}
&D_{\mathrm{KL}}(q_\phi(\mathbf{Y}^\tau|\mathbf{W}^\tau, \boldsymbol{\Omega}^\tau) \parallel p(\mathbf{Y}^\tau|\mathbf{W}^\tau, \boldsymbol{\Omega}^\tau)) \qquad (23)\\
=&D_{\mathrm{KL}}(q_\phi(\mathbf{Y}^\tau|\mathbf{W}^\tau, \mathbf{S}^\tau) \parallel p(\mathbf{Y}^\tau|\mathbf{W}^\tau, \mathbf{S}^\tau))\\
=&\log p(\mathbf{W}^\tau, \mathbf{S}^\tau) - \mathbb{E}_{q_\phi}[\log p(\mathbf{W}^\tau, \mathbf{S}^\tau|\mathbf{Y}^\tau)]\\
&+ D_{\mathrm{KL}}(q_\phi(\mathbf{Y}^\tau|\mathbf{W}^\tau, \mathbf{S}^\tau) \parallel p(\mathbf{Y}^\tau)),
\end{aligned}
$$

where the joint distribution is considered invariant. Note that the mapping from $(\mathbf{W}^\tau, \mathbf{S}^\tau)$ to $\mathbf{Y}^\tau$ is a bijection thus we omit the reconstruction term $\mathbb{E}_{q_\phi}[\log p(\mathbf{W}^\tau, \mathbf{S}^\tau|\mathbf{Y}^\tau)]$. Overall, Eq. (23) is simplified as:

$$
\begin{aligned}
&D_{\mathrm{KL}}(q_\phi(\mathbf{Y}^\tau|\mathbf{W}^\tau, \boldsymbol{\Omega}^\tau \parallel p(\mathbf{Y}^\tau|\mathbf{W}^\tau, \boldsymbol{\Omega}^\tau)) \qquad (24)\\
\simeq&D_{\mathrm{KL}}(q_\phi(\mathbf{Y}^\tau|\mathbf{W}^\tau, \mathbf{S}^\tau) \parallel p(\mathbf{Y}^\tau))\\
=&\int q_\phi(\mathbf{Y}^\tau|\mathbf{W}^\tau, \mathbf{S}^\tau) \log \frac{q_\phi(\mathbf{Y}^\tau|\mathbf{W}^\tau, \mathbf{S}^\tau)}{p(\mathbf{Y}^\tau)} d\mathbf{Y}^\tau\\
=&\int q_\phi(\mathbf{Y}^\tau|\mathbf{W}^\tau, \mathbf{S}^\tau) \log q_\phi(\mathbf{Y}^\tau|\mathbf{W}^\tau, \mathbf{S}^\tau) d\mathbf{Y}^\tau\\
&- \int q_\phi(\mathbf{Y}^\tau|\mathbf{W}^\tau, \mathbf{S}^\tau) \log p(\mathbf{Y}^\tau) d\mathbf{Y}^\tau\\
=&- H\left[q_\phi(\mathbf{Y}^\tau|\mathbf{W}^\tau, \mathbf{S}^\tau)\right] - \mathbb{E}_{q_\phi}\left[\log p(\mathbf{Y}^\tau)\right]\\
\simeq&\sum_{i=1}^{N} \left(p\left(y_i^\tau\right) \log p\left(y_i^\tau\right)\right) - \frac{1}{N}\sum_{i=1}^{N} \log \mathcal{N}\left(y_i|\mu_i^\tau, \left(\sigma_i^\tau\right)^2\right),
\end{aligned}
$$

where $\log p(y_i^\tau)$ is estimated by the Hutchinson's estimator, expectation $\mathbb{E}_{q_\phi}\left[\log p(\mathbf{Y}^\tau)\right]$ is estimated by Monte Carlo (Eq. (15)), and $H$ is the entropy. Besides, we assume the distributions of deformations at time $\tau$ and $\tau+1$ are the same. We minimize the above KL divergence by aggregating all the model predictions at different locations:

$$
\begin{aligned}
\mathcal{L} =&\frac{1}{N\Delta T}\sum_{\tau}^{T'}\sum_{i=1}^{N} \left(y_i^\tau - \hat{y}_i^\tau\right)^2 + \sum_{\tau}^{T'}\sum_{i=1}^{N} p\left(\hat{y}_i^\tau\right) \log p\left(\hat{y}_i^\tau\right)\\
&- \frac{1}{N}\sum_{\tau}^{T'}\sum_{i=1}^{N} \log \mathcal{N}\left(\hat{y}_i^\tau|\mu_i^\tau, \left(\sigma_i^\tau\right)^2\right), \qquad (25)
\end{aligned}
$$

where $\Delta T = T' - T$ is the forecasting horizon, $\tau = T + 1$, $y_i^\tau$ and $\hat{y}_i^\tau$ denote the ground truth and prediction of the $i$-th location at time $\tau$, respectively.

### *F. Training Details*

To remedy the situation that training neural networks often rely on extensive deformation data, we use simulated perturbations to pre-train $\mathbf{v}_i' = f^{-1}(\mathbf{u}_i|\mathbf{s}_i^\tau)$. Specifically, we split the studied area into $K$ grids and randomly sample a perturbed vector $\mathbf{r}_i^k \sim \mathcal{N}(\boldsymbol{\mu}^k, {\boldsymbol{\sigma}^k}^2)$ for the $k$-th gird, where $\boldsymbol{\mu}^k \sim \mathrm{unif}\left[\boldsymbol{\mu} - \boldsymbol{\alpha}, \boldsymbol{\mu} + \boldsymbol{\alpha}\right]$ and $\boldsymbol{\sigma}^k \sim \mathrm{unif}\left[\boldsymbol{\sigma} - \boldsymbol{\beta}_1, \boldsymbol{\sigma} + \boldsymbol{\beta}_2\right]$, $\boldsymbol{\beta}_1 > \boldsymbol{\beta}_2$. Then we add the simulated perturbations $\mathbf{r}_i^k$ to each monitored location $\mathbf{v}_i$ to simulate the deformation. Subsequently, the deformation distribution is estimated as $\mathbf{v}_i' = f^{-1}(\mathbf{u}_i|\mathbf{r}_i^k)$. Once the pre-training converged, we can use the real data to fine-tune $f$, obtaining a better mapping between data and manifold.

In addition, we propose a two-stage training strategy in DyLand – dynamic manifold learning and land deformation forecasting – rather than training them end-to-end. First, we argue that the manifold learning should be decoupled from the forecasting task since the manifold has its physical meaning. Second, the two-stage training is time- and space-efficient. By pre-training the manifold, we can free more GPU memories and have larger batches for the forecasting model.

## V. Experiments

In this section, we first describe the experimental settings including datasets, baselines, and evaluation metrics. We then

TABLE II
DESCRIPTIVE STATISTICS OF DATASETS.

| Dataset | HZY | PBG |
|---|---|---|
| time | 11/30/2018 - 09/08/2019 | 11/17/2017 - 01/04/2020 |
| # points | 4,569 (west) 2,164 (east) | 5,886 (west), 8,671 (east) |
| displacement | [-29.06, 30.50] | [-60.45, 110.57] |
| Long. (E102°) | [1'50", 3'46"] | [46'36", 54'17"] |
| Lat. (N29°) | [39'12", 41'25"] | [12'41", 15'50"] |
| elevation | [1470.2, 2899.6] | [661.3, 2101.0] |

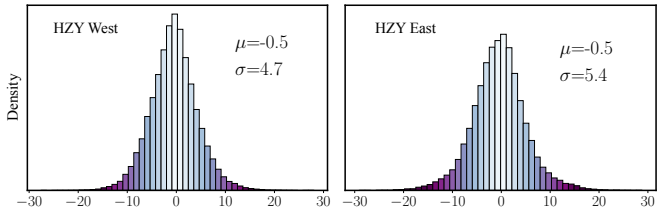



Fig. 4. Distribution of land deformations of the HZY datasets. We can see that the terrain surface deformations generally follow Gaussian distributions.

report the forecasting performance of DyLand compared with baselines. We also conduct ablation studies and discuss our main findings.

### A. Experimental Settings

*1) Datasets:* The studied areas used for evaluation are two real-world landslide-prone slopes in the southwest of Sichuan province, China, **HZY** and **PBG**. We collect InSAR measurements of the monitored slopes, observing surface deformations over time. The HZY data is composed by about nine months of slope observations on both sides of the *Dadu River*, between Nov 30, 2018, to Sept 8, 2019. The PBG data is recorded on a different place of the same river over two years of InSAR measurements (Nov 17, 2017, to Jan 04, 2020). Table II summarizes the statistics of the two studied areas. The land deformation distributions are shown in Fig. 4. The in-situ maps of the studies areas are shown in Fig. 5, where we have InSAR observation positions marked on the map.

*2) Hyper-parameters & settings:* We split the datasets into training (50%), validation (30%), and test (20%) sets. Geographical coordinates are max-min normalized. All deep learning methods are optimized by Adam optimizer with a learning rate of $10^{-3}$ and weight decay of $10^{-5}$. Early stopping is triggered with a patience of 100 epochs. All experiments are reported by the best result in 20 runs. The latent manifold $\mathbf{U}$ is 3D for evaluation and $2D$ for visualization. The dimension of spatial representation $\mathbf{W}$ is 5. The deformation for each location $\mathbf{s}_i^\tau$ is scaled up by multiplying 10 and then passed to NF for conditioning, since the scale of $\mathbf{s}_i^\tau$ is too small compared to $\mathbf{V}$. The NF and neural ODEs have 3 layers and a dimension of 64. The ODE Solver is dopri5. The number of hidden neurons in FC is equal to the input length $|\mathbf{S}|$. Both FC and GNNs are 3 layers.

*3) Baselines:* We compare DyLand with the following sixteen baselines. (i) **HA** calculates the historical data average of time period $T$ as a simple heuristic benchmark. We have $T = 1$ and $T = 3$ marked as HA(1) and HA(3), respectively; (ii) support vector regression (**SVR**) is a classic time-series forecasting model; (iii) **ARIMA** combines autoregressive and moving average for prediction; (iv) **GRU**, **LSTM**, and **Bi-LSTM** are recurrent neural networks that have been widely used for temporal forecasting of landslide displacement [27], [59]; (v) Transformer [60] is an attention-based model that achieves better performance than recurrent models on various language and vision tasks; (vi) **NODE** [51] is a continuous NF model that learns time-series by solving ODEs; (vii) **GCN** [61] is a classic GNN which models spatial dependencies via graph convolution; (viii) **VGAE** [55] is an unsupervised variational framework making use of latent variables learned from the GCN autoencoder; (ix) **SIG-VAE** [62] is similar to VGAE but has more flexible and complex posterior approximations; (x) **STGCN** [46] filters input and hidden states passed to a recurrent unit via graph convolutions; (xi) **P-GNN** [29] is a point cloud graph convolution method, we adapt it to our problem by adding GRU layers into it; (xii) **STGODE** [63] incorporates ordinary differential equations (ODEs) to capture long-term spatial-temporal features; (xiii) **SA-GNN** [25] learns surface manifold embeddings and uses GCNs and GRUs to model spatio-temporal dependencies. In Table IV, we list the model size and training time of DyLand and GNN baselines. The complexity of GNN baselines are mainly from the operations of graph convolution and dense layers in neural networks. For STGCN, it approximates graph kernels via Chebyshev polynomials and first-order approximations, which is as efficient as DyLand, and its complexity $\mathcal{O}(N)$ is also linear to the number of nodes $N$. SA-GNN adopts a similar architecture to STGCN, but with more spatial and temporal modules that require computation, resulting in longer training times. STODE uses ODEs to continuously model long-term spatial-temporal dependencies, which saves model parameters but is more time-consuming. As for P-GNN, its complexity is $\mathcal{O}(N \log^2 N)$, which is more expensive than other models. All models require less than an hour of training time with GPU acceleration, which are feasible in practical situations.

*4) Metrics:* We use five standard metrics for evaluating the forecasting performance. They are: RMSE, MAE, Accuracy (counted as correct if the forecasting error $|y_i^\tau - \hat{y}_i^\tau| \leq 0.01$), R Square, and explained variance score (EVS).

### B. Land Deformation Forecasting Results

We show the land deformation forecasting results in Table III, which reports the performance comparison between DyLand and baselines on real-world land deformation data in terms of five metrics. The best results are in bold, where we can see that our proposed DyLand outperforms baselines uniformly. We have following observations. First, baselines simply relying on temporal observations (i.e., they leave out spatial information of land surface) are not enough to accurately forecast the land deformations. In contrast, graph-based approaches generally achieve better results, which demonstrates that spatial relations and collective deformation trends are critical for forecasting. More in detail, we can see that P-GNN and SA-GNN perform better than other ST-GNNs since

TABLE III
OVERALL PERFORMANCE COMPARISONS BETWEEN DYLAND AND BASELINES ON LAND DEFORMATION PREDICTION.

| Dataset | HZY-West | | | | | HZY-East | | | | | PBG-West | | | | | PBG-East | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **RMSE** | **MAE** | **ACC** | **$R^2$** | **EVS** | **RMSE** | **MAE** | **ACC** | **$R^2$** | **EVS** | **RMSE** | **MAE** | **ACC** | **$R^2$** | **EVS** | **RMSE** | **MAE** | **ACC** | **$R^2$** | **EVS** |
| HA(1) | 4.190 | 2.799 | 0.072 | 0.054 | 0.161 | 4.973 | 3.427 | 0.056 | 0.065 | 0.178 | 8.348 | 5.114 | 0.044 | 0.062 | 0.120 | 6.944 | 4.194 | 0.046 | 0.061 | 0.117 |
| HA(3) | 3.992 | 2.889 | 0.052 | 0.066 | 0.122 | 4.645 | 3.352 | 0.045 | 0.094 | 0.158 | 7.535 | 5.186 | 0.040 | 0.092 | 0.142 | 6.067 | 4.010 | 0.050 | 0.097 | 0.164 |
| SVR | 3.956 | 3.209 | 0.030 | 0.108 | 0.274 | 4.750 | 3.693 | 0.032 | 0.135 | 0.262 | 6.612 | 4.199 | 0.059 | 0.195 | 0.291 | 5.955 | 3.546 | 0.076 | 0.191 | 0.282 |
| ARIMA | 4.888 | 4.088 | 0.053 | 0.265 | 0.215 | 6.704 | 5.395 | 0.031 | 0.271 | 0.204 | 6.242 | 4.098 | 0.035 | 0.190 | 0.212 | 5.325 | 3.200 | 0.055 | 0.175 | 0.228 |
| GRU | 0.250 | 0.204 | 0.421 | 0.223 | 0.315 | 0.197 | 0.170 | 0.456 | 0.207 | 0.290 | 0.249 | 0.197 | 0.460 | 0.120 | 0.163 | 0.200 | 0.160 | 0.540 | 0.215 | 0.137 |
| LSTM | 0.241 | 0.200 | 0.433 | 0.230 | 0.317 | 0.192 | 0.168 | 0.460 | 0.213 | 0.295 | 0.248 | 0.195 | 0.461 | 0.122 | 0.164 | 0.195 | 0.156 | 0.548 | 0.220 | 0.139 |
| Bi-LSTM | 0.240 | 0.199 | 0.435 | 0.230 | 0.318 | 0.190 | 0.165 | 0.464 | 0.216 | 0.299 | 0.248 | 0.195 | 0.461 | 0.123 | 0.164 | 0.193 | 0.152 | 0.552 | 0.224 | 0.141 |
| Transf. | 0.130 | 0.099 | 0.614 | 0.389 | 0.425 | 0.143 | 0.107 | 0.480 | 0.387 | 0.366 | 0.074 | 0.045 | 0.551 | 0.277 | 0.258 | 0.080 | 0.045 | 0.696 | 0.337 | 0.349 |
| NODE | 0.076 | 0.057 | 0.527 | 0.407 | 0.458 | 0.101 | 0.075 | 0.506 | 0.469 | 0.417 | 0.052 | 0.044 | 0.584 | 0.283 | 0.291 | 0.053 | 0.041 | 0.710 | 0.401 | 0.412 |
| GCN | 0.100 | 0.076 | 0.442 | 0.365 | 0.332 | 0.103 | 0.078 | 0.440 | 0.399 | 0.343 | 0.063 | 0.050 | 0.574 | 0.204 | 0.208 | 0.089 | 0.056 | 0.662 | 0.385 | 0.345 |
| VGAE | 0.091 | 0.068 | 0.483 | 0.390 | 0.391 | 0.104 | 0.077 | 0.465 | 0.392 | 0.313 | 0.051 | 0.040 | 0.601 | 0.375 | 0.332 | 0.083 | 0.052 | 0.712 | 0.355 | 0.348 |
| SIG-VAE | 0.088 | 0.065 | 0.525 | 0.374 | 0.422 | 0.096 | 0.071 | 0.513 | 0.499 | 0.440 | 0.045 | 0.037 | 0.734 | 0.405 | 0.426 | 0.079 | 0.050 | 0.775 | 0.408 | 0.393 |
| STGCN | 0.069 | 0.055 | 0.533 | 0.456 | 0.451 | 0.078 | 0.057 | 0.562 | 0.563 | 0.573 | 0.040 | 0.029 | 0.815 | 0.447 | 0.488 | 0.041 | 0.027 | 0.854 | 0.437 | 0.426 |
| P-GNN | 0.065 | 0.048 | 0.628 | 0.423 | 0.466 | 0.071 | 0.051 | 0.637 | 0.507 | 0.516 | 0.031 | 0.020 | 0.908 | 0.474 | 0.491 | 0.032 | 0.026 | 0.911 | 0.441 | 0.449 |
| STGODE | 0.063 | 0.046 | 0.645 | 0.459 | 0.471 | 0.069 | 0.051 | 0.642 | 0.505 | 0.515 | 0.029 | 0.019 | 0.911 | 0.478 | 0.494 | 0.030 | 0.024 | 0.916 | 0.449 | 0.453 |
| SA-GNN | 0.058 | 0.037 | 0.718 | 0.488 | 0.483 | 0.062 | 0.048 | 0.698 | 0.497 | 0.501 | 0.021 | 0.015 | 0.964 | 0.492 | 0.480 | 0.024 | 0.018 | 0.956 | 0.470 | 0.478 |
| **DyLand** | **0.052** | **0.035** | **0.742** | **0.492** | **0.492** | **0.060** | **0.043** | **0.700** | **0.583** | **0.590** | **0.013** | **0.010** | **0.993** | **0.538** | **0.540** | **0.016** | **0.013** | **0.978** | **0.487** | **0.496** |

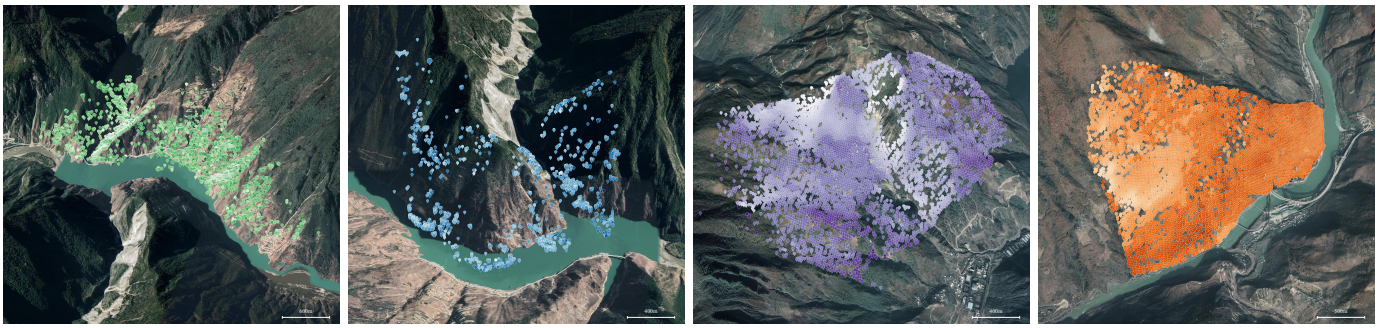


Fig. 5. Studied landslide-prone areas. InSAR observations are marked on the land surface, and their colors indicate deformation magnitude. From left to right: HZY-West, HZY-East, PBG-West, and PBG-East. Photos are obtained via Google Earth service.

TABLE IV
COMPARISON OF MODEL PARAMETERS AND TRAINING TIME OF DYLAND AND GNN-BASED BASELINES ON HZY-WEST.

| Method | Parameters | Time of 100 Iters |
|---|---|---|
| STGCN | 84,545 | 20.26 mins |
| P-GNN | 44,995 | 26.47 mins |
| STGODE | 43,058 | 28.71 mins |
| SA-GNN | 84,033 | 36.88 mins |
| DyLand | 18,341 | 15.96 mins |

they are able to capture the complex correlations on manifold. SA-GNN slightly outperforms P-GNN and STGODE, because the relative positions such as slope and azimuth are explicitly embedded, which can partially discriminate the importance of adjacent points.

When comparing DyLand with the best baseline SA-GNN, the former is capable of modeling manifold dynamics instead of the static surface embedding in the later. DyLand alleviates the bias caused by the surface deformation and considers the deformation learning and spatial representation as a dynamic system, which bridges the gap between separate "S" and "T" learning in previous ST-GNN models – as a result, it is more accurate for land deformation forecasting.

### C. Why Modeling Dynamic Manifold?

We now give our reasons for modeling dynamic manifold by presenting the visualization, quantitative comparison, as well as ablation studies of the co-training and hyper-parameters.

*1) Visualization of surface representations:* Dynamic manifold learning plays an important role in DyLand. We first provide manifold visualization of DyLand and eight representative manifold learning methods in the 2D space. They are LLE [26], MLLE [64], HLLE [34], LTSA [65], $t$-SNE [36], WLLE [25], Isomap [35], and UMAP [37]. Recall that DyLand learns the adjacency $p(\mathbf{A}|\mathbf{V},\mathbf{S})$ by explicitly considering the temporal land deformations $\mathbf{S}$. To ensure a fair comparison, we include $\mathbf{S}$ as another temporal channel in addition to the spatial 3D channels for other models. We visualize the original 3D land surfaces of PBG west riversides and then plot the 2D surface representations learned by different manifold methods in Fig. 6 colored by their geographical coordinates in the original 3D space. UMAP and $t$-SNE – as typical dimension reduction algorithms – over-scattered (clustered) the surface representations, and therefore lose continuous spatial correlations essential for deformation forecasting. Other

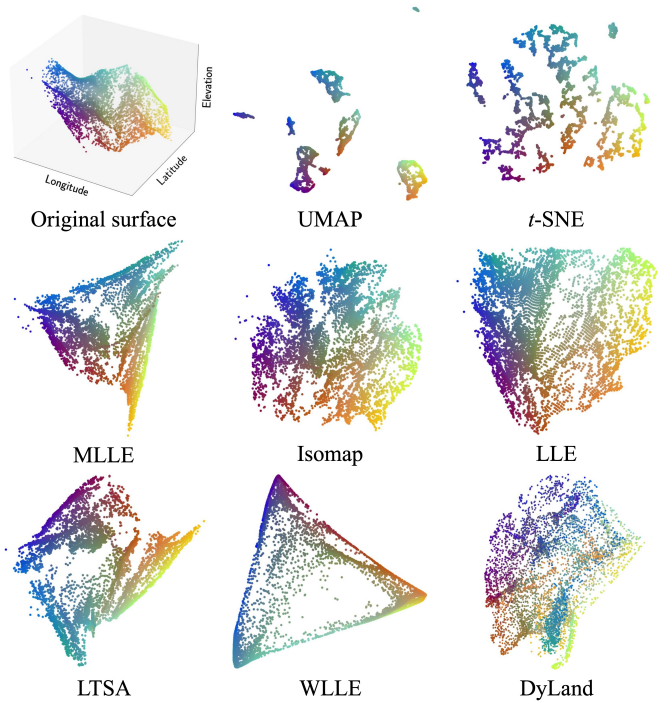



Fig. 6. Learned Land manifolds on PBG west riverside.

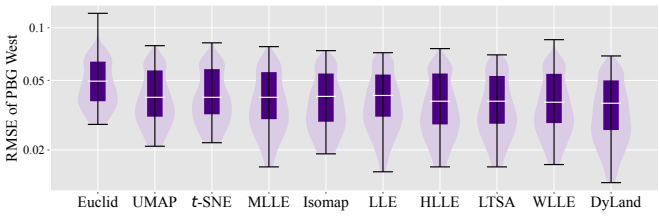



Fig. 7. Quantitative comparison between dynamic manifold (DyLand) and static manifold learning methods. We replace the dynamic manifold in DyLand with the nine alternatives and report the prediction results (RMSE) as violin plot. "Euclid" takes original surface as input directly. Box denotes the quartiles (25% and 75%) and the white line in the box is median value.

static manifold learning methods such as MLLE, Isomap, LLE, and LTSA can preserve the original manifold due to their ability to capture local Euclidean topology. However, they generally model the static manifold and cannot discriminate the influence of neighboring nodes. WLLE considers the slope and azimuth information but significantly distorts the latent representations where local manifold structure is disregarded. In addition to the subjective visualizations, we also provide a quantitative comparison between manifold learning methods since the most meaningful way to evaluate manifold representations is an actual downstream application.

*2) Quantitative comparison on manifold learning:* To quantify the benefits of jointly preserving the local manifold structure and temporal evolution of land deformations in DyLand, we replace the dynamic manifold learning module in DyLand with nine alternatives. We show the comparison results (measured by RMSE) in Fig. 7 as violin plot. We can see that DyLand has more miner prediction errors compared to other variants. This result validates our motivation by modeling

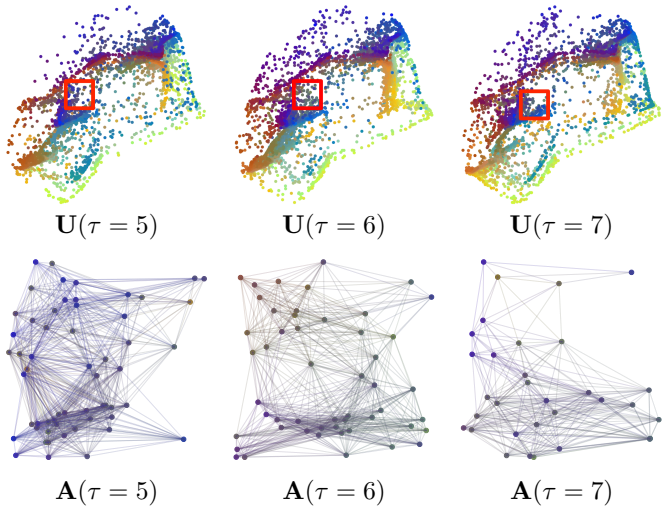



Fig. 8. Comparison of manifolds and sampled relevant adjacency matrix over time on PBG west riverside.

dynamic manifold surfaces in land deformation forecasting. Although the local topology patterns on the terrain surface can be preserved in other manifold methods, land deformations are not explicitly considered, resulting in fixed adjacency and thus determined neighboring feature aggregation (as they cannot capture the intricate correlations beyond Euclidean space).

*3) Effect of dynamic adjacency modeling:* To understand the dynamic adjacency modeling in DyLand intuitively, we plot the manifold embeddings at different timestamps in Fig. 8, as well as the sampled adjacency matrices (framed by the red rectangle in the top row of Fig. 8). The results show how embeddings and corresponding adjacency are changing over time, which is more obvious when observing the adjacency matrices. DyLand is able to learn dynamic manifold from deformations and construct the matrices accordingly (i.e., interactions among points), allowing us to better capture the point interactions adaptively without hyper-parameter tuning. In contrast, static manifold learning approaches are parameter-sensitive, e.g., presumed near neighbor specification. More importantly, DyLand generates adjacency $\mathbf{A}$ considering both geographical locations $\mathbf{V}$ and temporal deformation observations $\mathbf{S}$. This property enables the model to dynamically aggregate deformations for adaptive forecasting.

### D. Effect of Co-Training & Hyper-parameter $M$

Recall that we consider the evolution of spatial representations $\mathbf{W}$ and temporal land deformations $\mathbf{S}$ as a dynamic system solved by neural ODEs, where $\mathbf{W} \in \mathbb{R}^{N\times M}$ is learned by GNNs. Now we investigate the effect of co-training strategy on land deformation forecasting. First, we introduce two variants of DyLand: (i) DyLand-$\mathbf{S}$, which removes the learning of $\mathbf{W}$ and makes deformation forecasting only based on temporal deformation observations, i.e., it degenerates to a typical ODE-based time-series method [51] through modeling $p(\mathbf{Y}|\mathbf{S})$; (ii) DyLand-$\mathbf{W}$, which removes the learning of $\mathbf{S}$ and only utilizes spatial representation for forecasting, i.e., $p(\mathbf{Y}|\mathbf{W})$, similar to existing ST-GNNs. As illustrated in Table V, modeling

TABLE V
EFFECT OF CO-TRAINING IN DYLAND.

| Model | RMSE | MAE | ACC | $R^2$ | EVS |
|---|---|---|---|---|---|
| *PBG West Riverside* | | | | | |
| DyLand-**S** | 0.052 | 0.044 | 0.584 | 0.283 | 0.291 |
| DyLand-**W** | 0.019 | 0.015 | 0.983 | 0.497 | 0.500 |
| DyLand | **0.013** | **0.010** | **0.993** | **0.538** | **0.540** |
| *PBG East Riverside* | | | | | |
| DyLand-**S** | 0.053 | 0.041 | 0.710 | 0.401 | 0.412 |
| DyLand-**W** | 0.023 | 0.018 | 0.965 | 0.452 | 0.489 |
| DyLand | **0.016** | **0.013** | **0.978** | **0.487** | **0.496** |
| *HZY West Riverside* | | | | | |
| DyLand-**S** | 0.076 | 0.057 | 0.527 | 0.407 | 0.458 |
| DyLand-**W** | 0.064 | 0.045 | 0.684 | 0.482 | 0.464 |
| DyLand | **0.052** | **0.035** | **0.742** | **0.492** | **0.492** |
| *HZY East Riverside* | | | | | |
| DyLand-**S** | 0.101 | 0.075 | 0.506 | 0.469 | 0.417 |
| DyLand-**W** | 0.063 | 0.045 | 0.671 | 0.491 | 0.497 |
| DyLand | **0.060** | **0.043** | **0.700** | **0.583** | **0.590** |

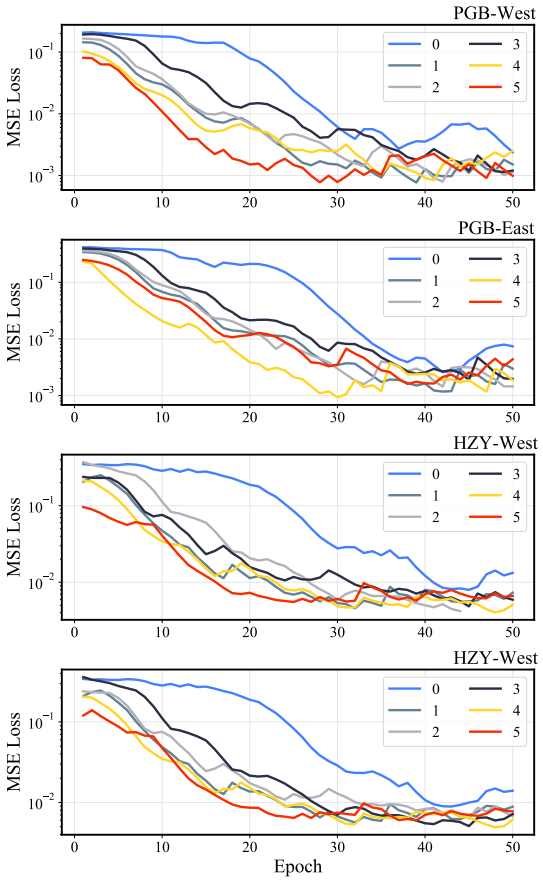



Fig. 9. Effect of hyper-parameter $M$ (from 0 to 5) during the first 50 epochs of model training. Y-axis is MSE loss.

the co-evolution of **W** and **S** as a holistic dynamic system significantly improves the forecasting performance. DyLand-**S** performs the worst due to ignoring the important spatial correlations, although it also uses neural ODEs for learning deformation dynamics. Nevertheless, incorporating **S** into DyLand-**W** on HZY dataset provides a lot of improvements, e.g., the accuracy are boosted from 68.4% to 74.2% on HZY-West and from 67.1% to 70.0% on HZY-East.

Co-training **S** and **W** with neural ODEs can be seen as an augmentation to the dynamic system **S** since the spatial representations **W** are also conditioned on the evolution of land deformations **S**. In other words, it solves the ODEs regarding **S** in an augmented space, which lifts the points into an additional space for smoother learning. Thus, the dimension $M$ of **W** is an important hyper-parameter, as it controls the output shape of spatial-temporal representation learning that should be preserved for forecasting.

Fig. 9 plots the training process of DyLand by varying the values of $M$ from 0 to 5. Note that $M = 0$ means our model only uses **S** for forecasting. Increasing $M$ can explore higher additional space of **W** to capture more complex dependencies and mappings while preserving the point cloud topology. Although larger $M$ generally makes the model converge fast and has fewer function evaluations, it is restricted by the dimension of **S**. Also, a larger $M$ (e.g., 5) makes the model unstable, which might be caused by the increasing complexity of functions required to be approximated in ODEs.

### E. Influence of the Priors

At last, we investigate the influence of different priors of spatial representation **W**, e.g., $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$, $\mathcal{N}(\mathbf{0}, \mathbf{I}^2)$, and no prior (i.e., we replace the VGAE with a deterministic GCN). The results are shown in Fig. 10, where the averaged standard deviation $\overline{\boldsymbol{\sigma}_{\mathbf{W}}} \simeq 1.2$ when approximating $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ and $\overline{\boldsymbol{\sigma}_{\mathbf{W}}} \simeq 0.9$ when approximating $\mathcal{N}(\mathbf{0}, \mathbf{I}^2)$ – both of which are significant smaller than the ground truth (cf. Fig. 4). The mean $\overline{\boldsymbol{\mu}_{\mathbf{W}}} \in [-0.05, 0.05]$ in both cases. Compared to GCN model, VGAE achieves better performance, which verifies our motivation that constraining the distribution of representations helps land deformation forecasting. We also find that a specially designed prior is beneficial for model stability, but does not show significant superiority on optimal results, as observed by similar values of $\boldsymbol{\mu}_{\mathbf{W}}$ and larger values of $\boldsymbol{\sigma}_{\mathbf{W}}$. While encouraging **W** to follow a normal distribution with a large standard deviation, the deviation is still smaller than expected due to the under-fitting and over-smoothing problems introduced by VGAE when fitting other parameters.

### F. Discussion

DyLand learns the spatial connections and temporal dependencies of land surface InSAR point cloud and combines them in unison. In addition to its capability of learning spatial-temporal land deformation dynamics, we also discuss DyLand's limitations that are worth further research. First, the point cloud data we used were collected from landslide-prone slopes of *Dadu River*. Other kinds of landslide-prone lands, e.g., coastal areas, can be used to evaluate the generalization ability of our model. Second, the distributions of land deformations in our dataset generally follow Gaussian distribution. New priors may be needed to accurately approximate spatial and temporal representations for areas with different deformation distributions. Third, many meteorological and edaphic

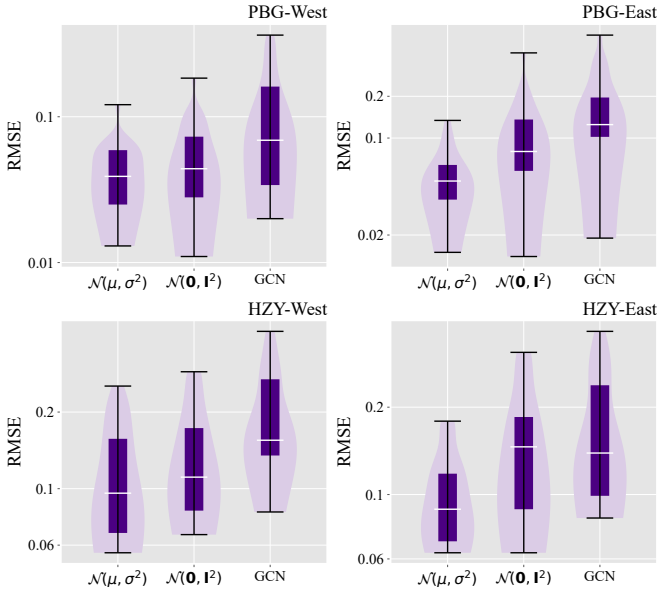



Fig. 10. Influence of the prior of $\mathbf{W}$. Y-axis is RMSE.

factors of the studied areas can be incorporated into our model to enhance the performance of land deformation prediction.

## VI. Conclusion

We presented DyLand, a general manifold learning model with application to forecasting land deformations, addressing the fundamental limitations of existing approaches in learning the spatio-temporal characteristics of InSAR point cloud data. Our novel dynamic manifold model learns deformation distribution on the terrain surface with tractable density estimation, preserving both topological structures and deformation sensitivity for highly accurate land deformation forecasting. DyLand also generalizes the dynamic process of learning temporal deformation and spatial representation through solving the ODEs of the co-evolution dynamic system. Extensive evaluations on real-world data demonstrated the superior performance of DyLand over existing spatio-temporal GNNs and static manifold methods. In our future work, we will attempt to extend DyLand by incorporating meteorological data and meta-learning.

## References

[1] F. Dai, C. Lee, and Y. Y. Ngai, "Landslide risk assessment and management: an overview," *Engineering Geology*, vol. 64, no. 1, pp. 65–87, 2002.

[2] F. V. de Blasio, *Introduction to Physics of Landslides*, 1st ed. Springer Dordrecht, 2020.

[3] S. Nang and R. C. Paddock, "Myanmar Jade Mine Collapse Kills at Least 168," The New York Times, Jul. 2, 2020. [Online]. Available: https://www.nytimes.com/2020/07/02/world/asia/myanmar-jade-mine-collapse.html

[4] M. Schlögl, K. Gutjahr, and S. Fuchs, "The challenge to use multi-temporal InSAR for landslide early warning," *Natural Hazards*, vol. 112, no. 3, pp. 2913–2919, 2022.

[5] E. Intrieri, G. Gigli, F. Mugnai, R. Fanti, and N. Casagli, "Design and implementation of a landslide early warning system," *Engineering Geology*, vol. 147, pp. 124–136, 2012.

[6] M. Devara, A. Tiwari, and R. Dwivedi, "Landslide susceptibility mapping using mt-InSAR and AHP enabled GIS-based multi-criteria decision analysis," *Geomatics, Natural Hazards and Risk*, vol. 12, no. 1, pp. 675–693, 2021.

[7] J. Dong, L. Zhang, M. Liao, and J. Gong, "Improved correction of seasonal tropospheric delay in InSAR observations for landslide deformation monitoring," *Remote Sensing of Environment*, vol. 233, p. 111370, 2019.

[8] A. Hajimoradlou, G. Roberti, and D. Poole, "Predicting landslides using locally aligned convolutional neural networks," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2020, pp. 3342–3348.

[9] Y. Dong, T. Liang, Y. Zhang, and B. Du, "Spectral–spatial weighted kernel manifold embedded distribution alignment for remote sensing image classification," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3185–3197, 2021.

[10] Y. Thiery, O. Maquaire, and M. Fressard, "Application of expert rules in indirect approaches for landslide susceptibility assessment," *Landslides*, vol. 11, no. 3, pp. 411–424, 2014.

[11] A.-X. Zhu, R. Wang, J. Qiao, C.-Z. Qin, Y. Chen, J. Liu, F. Du, Y. Lin, and T. Zhu, "An expert knowledge-based approach to landslide susceptibility mapping using GIS and fuzzy logic," *Geomorphology*, vol. 214, pp. 128–138, 2014.

[12] S. Wan, T.-C. Lei, and T.-Y. Chou, "A landslide expert system: Image classification through integration of data mining approaches for multi-category analysis," *International Journal of Geographical Information Science (IJGIS)*, vol. 26, no. 4, pp. 747–770, 2012.

[13] S. Steger, V. Mair, C. Kofler, M. Pittore, M. Zebisch, and S. Schneiderbauer, "Correlation does not imply geomorphic causation in data-driven landslide susceptibility modelling–benefits of exploring landslide data collection effects," *Science of the Total Environment*, vol. 776, p. 145935, 2021.

[14] B.-R. Gan, X.-G. Yang, and J.-W. Zhou, "GIS-based remote sensing analysis of the spatial-temporal evolution of landslides in a hydropower reservoir in southwest China," *Geomatics, Natural Hazards and Risk*, vol. 10, no. 1, pp. 2291–2312, 2019.

[15] T. Carlà, E. Intrieri, F. Raspini, F. Bardi, P. Farina, A. Ferretti, D. Colombo, F. Novali, and N. Casagli, "Perspectives on the prediction of catastrophic slope failures from satellite InSAR," *Scientific Reports*, vol. 9, p. 14137, 2019.

[16] P. Kayastha, M. R. Dhital, and F. De Smedt, "Application of the analytical hierarchy process (AHP) for landslide susceptibility mapping: A case study from the Tinau watershed, west Nepal," *Computers & Geosciences*, vol. 52, pp. 398–408, 2013.

[17] A. Shirzadi, D. T. Bui, B. T. Pham, K. Solaimani, K. Chapi, A. Kavian, H. Shahabi, and I. Revhaug, "Shallow landslide susceptibility assessment using a novel hybrid intelligence approach," *Environmental Earth Science*, vol. 76, no. 2, pp. 1–18, 2017.

[18] B. Kalantar, B. Pradhan, S. A. Naghibi, A. Motevalli, and S. Mansor, "Assessment of the effects of training data selection on the landslide susceptibility mapping: a comparison between support vector machine (SVM), logistic regression (LR) and artificial neural networks (ANN)," *Geomatics, Natural Hazards and Risk*, vol. 9, no. 1, pp. 49–69, 2018.

[19] W. Chen, X. Xie, J. Peng, J. Wang, Z. Duan, and H. Hong, "GIS-based landslide susceptibility modelling: a comparative assessment of kernel logistic regression, naïve-bayes tree, and alternating decision tree models," *Geomatics, Natural Hazards and Risk*, vol. 8, no. 2, pp. 950–973, 2017.

[20] H. Hong, B. Pradhan, M. N. Jebur, D. T. Bui, C. Xu, and A. Akgun, "Spatial prediction of landslide hazard at the Luxi area (China) using support vector machines," *Environmental Earth Science*, vol. 75, no. 1, pp. 1–14, 2016.

[21] M. C. Gemici, D. Rezende, and S. Mohamed, "Normalizing flows on riemannian manifolds," *arXiv:1611.02304*, pp. 1–3, 2016.

[22] D. J. Rezende, G. Papamakarios, S. Racaniere, M. Albergo, G. Kanwar, P. Shanahan, and K. Cranmer, "Normalizing flows on tori and spheres," in *International Conference on Machine Learning (ICML)*, 2020, pp. 8083–8092.

[23] J. Brehmer and K. Cranmer, "Flows for simultaneous manifold learning and density estimation," in *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020, pp. 442–453.

[24] T. Lei, Y. Zhang, Z. Lv, S. Li, S. Liu, and A. K. Nandi, "Landslide inventory mapping from bitemporal images using deep convolutional neural networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 6, pp. 982–986, 2019.

[25] F. Zhou, R. Li, K. Zhang, and G. Trajcevski, "Land deformation prediction via slope-aware graph neural networks," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2021, pp. 15 033–15 040.

[26] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[27] Z. Lin, X. Sun, and Y. Ji, "Landslide displacement prediction based on time series analysis and double-bilstm model," *International Journal of Environmental Research and Public Health*, vol. 19, no. 4, p. 2077, 2022.

[28] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics*, vol. 38, no. 5, pp. 1–12, 2019.

[29] W. Shi and R. Rajkumar, "Point-GNN: Graph neural network for 3D object detection in a point cloud," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 1711–1719.

[30] F. Zhou, R. Li, Q. Gao, G. Trajcevski, K. Zhang, and T. Zhong, "Dynamic manifold learning for land deformation forecasting," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2022.

[31] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference," *Journal of Machine Learning Research*, vol. 22, no. 57, pp. 1–64, 2021.

[32] R. Ran, J. Feng, S. Zhang, and B. Fang, "A general matrix function dimensionality reduction framework and extension for manifold learning," *IEEE Transactions on Cybernetics*, vol. 52, no. 4, pp. 2137–2148, 2022.

[33] M. Lei, P. Quan, R. Ma, Y. Shi, and L. Niu, "DigGCN: Learning compact graph convolutional networks via diffusion aggregation," *IEEE Transactions on Cybernetics*, vol. 52, no. 2, pp. 912–924, 2022.

[34] D. L. Donoho and C. Grimes, "Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data," *Proceedings of the National Academy of Sciences*, vol. 100, no. 10, pp. 5591–5596, 2003.

[35] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[36] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[37] L. McInnes, J. Healy, N. Saul, and L. Großberger, "UMAP: Uniform manifold approximation and projection," *Journal of Open Source Software*, vol. 3, no. 29, p. 861, 2018.

[38] H. Kim, H. Lee, W. H. Kang, J. Y. Lee, and N. S. Kim, "Softflow: Probabilistic framework for normalizing flow on manifolds," in *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020, pp. 16 388–16 397.

[39] M. Kuznetsov and D. Polykovskiy, "MolGrow: A graph normalizing flow for hierarchical molecular generation," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2021, pp. 8226–8234.

[40] S. Deutsch, A. Ortega, and G. Medioni, "Manifold denoising based on spectral graph wavelets," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 4673–4677.

[41] C.-J. F. Chung, A. G. Fabbri *et al.*, "Probabilistic prediction models for landslide hazard mapping," *Photogrammetric Engineering and Remote Sensing*, vol. 65, no. 12, pp. 1389–1399, 1999.

[42] A. Ding, Q. Zhang, X. Zhou, and B. Dai, "Automatic recognition of landslide based on cnn and texture change detection," in *Youth Academic Annual Conference of Chinese Association of Automation*, 2016, pp. 444–448.

[43] K. A. H. Kelany, A. Baniasadi, N. Dimopoulos, and M. Gara, "Improving InSAR image quality and co-registration through CNN-based super-resolution," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 1–5.

[44] C. Tian, X. Zhang, J. C.-W. Lin, W. Zuo, Y. Zhang, and C.-W. Lin, "Generative adversarial networks for image super-resolution: A survey," *arXiv:2204.13620*, 2022.

[45] C. Tian, Y. Zhang, W. Zuo, C.-W. Lin, D. Zhang, and Y. Yuan, "A heterogeneous group CNN for image super-resolution," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[46] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, 2020.

[47] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *International Conference on Learning Representations (ICLR)*, 2018, pp. 1–18.

[48] X. Wang, Y. Ma, Y. Wang, W. Jin, X. Wang, J. Tang, C. Jia, and J. Yu, "Traffic flow prediction via spatial temporal graph neural network," in *The Web Conference*, 2020, pp. 1082–1092.

[49] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2018, pp. 3634–3640.

[50] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848 – 3858, 2019.

[51] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2018, pp. 1–13.

[52] C. Chen, C. Li, L. Chen, W. Wang, Y. Pu, and L. C. Duke, "Continuous-time flows for efficient inference and density estimation," in *International Conference on Machine Learning (ICML)*, 2018, pp. 824–833.

[53] W. Grathwohl, R. T. Chen, J. Bettencourt, I. Sutskever, and D. Duvenaud, "FFJORD: Free-form continuous dynamics for scalable reversible generative models," in *International Conference on Learning Representations (ICLR)*, 2018, pp. 1–13.

[54] M. F. Hutchinson, "A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines," *Communications in Statistics-Simulation and Computation*, vol. 18, no. 3, pp. 1059–1076, 1989.

[55] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv:1611.07308*, pp. 1–3, 2016.

[56] L. Zhao and L. Akoglu, "PairNorm: Tackling oversmoothing in GNNs," in *International Conference on Learning Representations (ICLR)*, 2020.

[57] J. R. Dormand and P. J. Prince, "A family of embedded Runge-Kutta formulae," *Journal of Computational and Applied Mathematics*, vol. 6, no. 1, pp. 19–26, 1980.

[58] D. J. MacKay, *Information theory, inference and learning algorithms*. Cambridge University Press, 2003.

[59] Y. Wang, H. Tang, J. Huang, T. Wen, J. Ma, and J. Zhang, "A comparative study of different machine learning methods for reservoir landslide displacement prediction," *Engineering Geology*, vol. 298, p. 106544, 2022.

[60] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Annual Conference on Neural Information Processing Systems (NIPS)*, vol. 30, pp. 1–11, 2017.

[61] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017, pp. 1–14.

[62] A. Hasanzadeh, E. Hajiramezanali, K. Narayanan, N. Duffield, M. Zhou, and X. Qian, "Semi-implicit graph variational auto-encoders," in *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2019, pp. 1–12.

[63] Z. Fang, Q. Long, G. Song, and K. Xie, "Spatial-temporal graph ode networks for traffic flow forecasting," in *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2021, pp. 364–373.

[64] Z. Zhang and J. Wang, "MLLE: Modified locally linear embedding using multiple weights," in *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2007, pp. 1593–1600.

[65] Z. Zhang and H. Zha, "Principal manifolds and nonlinear dimensionality reduction via tangent space alignment," *SIAM Journal on Scientific Computing*, vol. 26, no. 1, pp. 313–338, 2004.

**Xovee Xu** (Graduate Student Member, IEEE) received the B.S. degree and M.S. degree in software engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, Sichuan, China, in 2018 and 2021, respectively, where he is currently pursuing the Ph.D. degree in computer science.

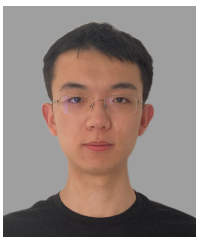
His research focuses on understanding spatial-temporal data, information diffusion, user-generated content, and human behaviors in social network.

**Ting Zhong** received the B.S. degree in computer application and the M.S. degree in computer software and theory from Beijing Normal University, Beijing, China, in 1999 and 2002, respectively, and the Ph.D. degree in information and communication engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2009.

She is a Full Professor with UESTC. Her current research interests include deep learning, social networks, and cloud computing.

**Fan Zhou** (Member, IEEE) received the B.S. degree in computer science from Sichuan University, Chengdu, China, in 2003, and the M.S. and Ph.D. degrees from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2006 and 2012, respectively.

He is currently a Professor with School of Information and Software Engineering, UESTC. His research interests include machine learning, neural networks, spatio-temporal data management, graph learning, recommender systems, and social network mining.



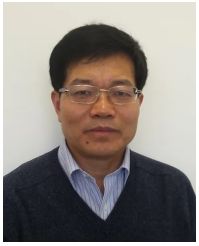
**Rongfan Li** received the B.S. degree and M.S. degree in software engineering from University of Electronic Science and Technology of China, in 2020 and 2023, respectively.

His research interests include machine learning, graph neural networks, spatio-temporal data mining, metric learning, representation learning.

**Goce Trajcevski** (Member, IEEE) received the B.Sc. degree in informatics and automation from the University of Sts. Kiril i Metodij, Skopje, North Macedonia, in 1989, and the M.S. and Ph.D. degrees in computer science from the University of Illinois at Chicago, Chicago, IL, USA, in 1995 and 2002, respectively.

He is currently a Kingland Associate Professor with the Department of Electrical and Computer Engineering, Iowa State University, Ames, IA, USA. His research has been funded by the NSF, ONR, BEA, and Northrop Grumman Corporation. In addition to a book chapter and three encyclopedia chapters, he has coauthored over 220 publications in refereed conferences and journals. His main research interests are in the areas of spatiotemporal data management, uncertainty and reactive behavior management in different application settings, and incorporating multiple contexts.

Dr. Trajcevski was the General Co-Chair of the IEEE International Conference on Data Engineering 2014 and ACM SIGSPATIAL 2019, the PC Co-Chair of the ADBIS 2018, ACM SIGSPATIAL 2016 and 2017, and IEEE MDM 2023. He has served in various roles in organizing committees in numerous conferences and workshops. He is an Associate Editor of the *ACM Transactions on Spatial Algorithms and Systems* and the *Geoinformatica* Journals.



**Qinggang Meng** (Senior Member, IEEE) received the B.S. degree in electronic engineering and the M.S. degree in signal and image processing from the School of Electronic Information Engineering, Tianjin University, Tianjin, China, in 1987 and 1990, respectively, and the Ph.D. degree in computer science from the University of Wales, Aberystwyth, U.K., in 2002.

He is currently a Professor of Robotics and AI with the Department of Computer Science, Loughborough University, Loughborough, U.K. His research interests include biologically inspired learning algorithms and developmental robotics, service robotics, agricultural robotics, driverless vehicles, robot learning and adaptation, multi-UAV cooperation, AI for Net Zero, digital twin, AI adversarial robustness, artificial intelligence, deep learning, computer vision, and embedded intelligence.

Prof. Meng is currently an associate editor for the IEEE TRANSACTIONS ON CYBERNETICS.