# A Survey on Graph Neural Networks for Time Series: Forecasting, Classification, Imputation, and Anomaly Detection

Ming Jin, Huan Yee Koh, Qingsong Wen, Daniele Zambon, Cesare Alippi, *Fellow, IEEE*, Geoffrey I. Webb, *Fellow, IEEE*, Irwin King, *Fellow, IEEE*, and Shirui Pan, *Senior Member, IEEE*

*(Survey Paper)*

***Abstract*—Time series are the primary data type used to record dynamic system measurements and generated in great volume by both physical sensors and online processes (virtual sensors). Time series analytics is therefore crucial to unlocking the wealth of information implicit in available data. With the recent advancements in graph neural networks (GNNs), there has been a surge in GNN-based approaches for time series analysis. These approaches can explicitly model inter-temporal and inter-variable relationships, which traditional and other deep neural network-based methods struggle to do. In this survey, we provide a comprehensive review of graph neural networks for time series analysis (GNN4TS), encompassing four fundamental dimensions: forecasting, classification, anomaly detection, and imputation. Our aim is to guide designers and practitioners to understand, build applications, and advance research of GNN4TS. At first, we provide a comprehensive task-oriented taxonomy of GNN4TS. Then, we present and discuss representative research works and introduce mainstream applications of GNN4TS. A comprehensive discussion of potential future research directions completes the survey. This survey, for the first time, brings together a vast array of knowledge on GNN-based time series research, highlighting foundations, practical applications, and opportunities of graph neural networks for time series analysis.**

***Index Terms*—Time series, graph neural networks, deep learning, forecasting, classification, imputation, anomaly detection.**

Manuscript received 17 December 2023; revised 8 July 2024; accepted 8 August 2024. Date of publication 14 August 2024; date of current version 5 November 2024. This work was supported in part by the CSIRO– National Science Foundation (US) AI Research Collaboration Program and Swiss National Science Foundation under Grant 204061. The work of Shirui Pan was supported in part by the Australian Research Council (ARC) under Grant FT210100097 and Grant DP240101547. Recommended for acceptance by W. Liu. *(Ming Jin and Huan Yee Koh contributed equally to this work.) (Corresponding author: Shirui Pan.)*

Ming Jin and Shirui Pan are with the School of Information and Communication Technology, Griffith University, Nathan, QLD 4111, Australia (e-mail: mingjinedu@gmail.com; s.pan@griffith.edu.au).

Huan Yee Koh and Geoffrey I. Webb are with the Department of Data Science and AI, Monash University, Clayton, VIC 3800, Australia (e-mail: huan.koh@monash.edu; geoff.webb@monash.edu).

Qingsong Wen is with Squirrel AI Learning, Bellevue, WA 98004 USA (e-mail: qingsongedu@gmail.com).

Daniele Zambon is with the Swiss AI Lab IDSIA, Università della Svizzera Italiana, 6900 Lugano, Switzerland (e-mail: daniele.zambon@usi.ch).

Cesare Alippi is with the Swiss AI Lab IDSIA, Università della Svizzera Italiana, 6900 Lugano, Switzerland, and also with Politecnico di Milano, 20133 Milano, Italy (e-mail: cesare.alippi@usi.ch).

Irwin King is with the Department of Computer Science & Engineering, Chinese University of Hong Kong, Ma Liu Shui, Hong Kong (e-mail: king@cse.cuhk.edu.hk).

This article has supplementary downloadable material available at https://doi.org/10.1109/TPAMI.2024.3443141, provided by the authors.

Digital Object Identifier 10.1109/TPAMI.2024.3443141

## I. Introduction

THE advent of advanced sensing and data stream processing technologies has led to an explosion of time series data [1], [2], [3]. The analysis of time series not only provides insights into past trends but also facilitates a multitude of tasks such as forecasting [4], classification [5], anomaly detection [6], and data imputation [7]. This lays the groundwork for time series modeling paradigms that leverage on historical data to understand current and future possibilities. Time series analytics have become increasingly crucial in various fields, including but not limited to cloud computing, transportation, energy, finance, social networks, and the Internet-of-Things [8], [9], [10].

Many time series involve complex interactions across time (such as lags in propagation of effects) and variables (such as the relationship among the variables representing neighboring traffic sensors). By treating time points or variables as nodes and their relationships as edges, a model structured in the manner of a network or graph can effectively solve the task at hand by exploiting both data and relational information. Indeed, much time series data is spatial-temporal in nature, with different variables in the series capturing information about different locations – space – too, meaning it encapsulates not only time information but also spatial relationships [11]. This is particularly evident in scenarios such as urban traffic networks, population migration, and global weather forecasting. In these instances, a localized change, such as a traffic accident at an intersection, an epidemic outbreak in a suburb, or extreme weather in a specific area, can propagate and influence neighboring regions. This spatial-temporal characteristic is a common feature of many dynamic systems, including the wind farm in Fig. 1, where the underlying time series displays a range of correlations and heterogeneities [15], [18]. Traditional analytic tools, such as support vector regression (SVR) [19], gradient boosting decision tree (GBDT) [20], vector autoregressive (VAR) [21], and autoregressive integrated moving average (ARIMA) [22], struggle to handle complex time series relations (e.g., nonlinearities and inter-variable relationships), resulting in less accurate prediction

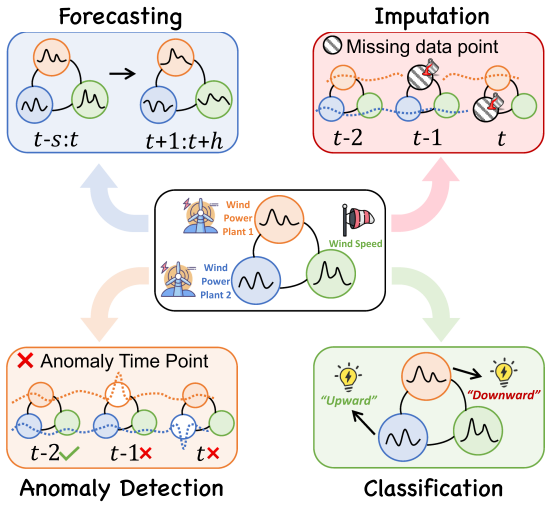



Fig. 1. Graph neural networks for time series analysis (GNN4TS). In this example of a wind farm, different analytical tasks can be categorized into time series forecasting, classification, anomaly detection, and imputation.

results [23]. The advent of deep learning technologies has led to the development of different neural networks based on convolutional neural networks (CNN) [24], [25], recurrent neural networks (RNN) [26], and transformers [27], which have shown significant advantages in modeling real-world time series data. However, one of the biggest limitations of the above methods is that they do not explicitly model the spatial relations existing between time series in non-euclidean space [15], which limits their expressiveness [28].

In recent years, graph neural networks (GNNs) have emerged as a powerful tool for learning non-euclidean data representations [29], [30], [31], [32], paving the way for modeling real-world time series data. This enables the capture of diverse and intricate relationships, both inter-variable (connections between different variables within a multivariate series) and inter-temporal (dependencies between different points in time). Considering the complex spatial-temporal dependencies inherent in real-world scenarios, a line of studies has integrated GNNs with various temporal modeling frameworks to capture both spatial and temporal dynamics and demonstrate promising results [12], [13], [14], [15], [16]. While early research efforts were primarily concentrated on various forecasting scenarios [13], [14], [15], recent advancements in time series analysis utilizing GNNs have demonstrated promising outcomes in other mainstream tasks. These include classification [33], [34], anomaly detection [35], [36], and imputation [37], [38]. In Fig. 1, we provide an overview of graph neural networks for time series analysis (GNN4TS).

*Related Surveys.* Despite the growing body of research performing various time series analytic tasks with GNNs, existing surveys tend to focus on specific perspectives within a restricted scope. For instance, the survey by Wang et al. [11] offers a review of deep learning techniques for spatial-temporal data mining, but it does not specifically concentrate on GNN-based methods. The survey by Ye et al. [12] zeroes in on graph-based deep learning architectures in the traffic domain, primarily considering forecasting scenarios. A recent survey by Jin et al. [15] offers an overview of GNNs for predictive learning in urban computing, but neither extends its coverage to other application domains nor thoroughly discusses other tasks related to time series analysis. Finally, we mention the work by Rahmani et al. [17], which expands the survey of GNNs to many intelligent transportation systems, but tasks other than forecasting remain overlooked. A detailed comparison between our survey and others is presented in Table I.

To fill the gap, this survey offers a comprehensive and up-to-date review of graph neural networks for time series analysis, encompassing the majority of tasks ranging from time series forecasting, classification, anomaly detection, and imputation. Specifically, we first provide two broad views to classify and discuss existing works from the task- and methodology-oriented perspectives. Then, we delve into six popular application sectors within the existing research of GNN4TS, and propose several potential future research directions. The key contributions of our survey are summarized as follows:

- *First Comprehensive Survey:* To the best of our knowledge, this is the first comprehensive survey that reviews the recent advances in mainstream time series analysis tasks with graph neural networks. It covers a wide range of recent research and provides a broad view of the development of GNN4TS without restricting to specific tasks or domains.
- *Unified and Structured Taxonomy:* We present a unified framework to structurally categorize existing works from task- and methodology-oriented perspectives. In the first classification, we offer an overview of tasks in time series analysis, covering different problem settings prevalent in GNN-based research; and in the second classification, we dissect GNN4TS in terms of spatial and temporal dependencies modeling and the overall model architecture.
- *Detailed and Current Overview:* We conduct a comprehensive review that not only covers the breadth of the field but also delves into the depth of individual studies with fine-grained classification and detailed discussion, providing readers with an up-to-date understanding of the state-of-the-art in GNN4TS.
- *Broadening Applications:* We discuss the expanding applications of GNN4TS across various sectors, highlighting its versatility and potential for future growth in diverse fields.
- *Future Research Directions:* We shed light on potential future research directions, offering insights and suggestions that could guide and inspire future research in the field of GNN4TS.

The remainder of this survey is organized as follows: Section II provides notations used throughout the paper. Section III presents the taxonomy of GNN4TS from different perspectives. Sections IV, V, VI, and VII review the four major tasks in the GNN4TS literature. Section VIII surveys popular applications of GNN4TS across various fields, while Section IX examines open questions and potential future directions.

## II. Definition and Notation

Time series data comprises a sequence of observations gathered or recorded over a period of time. This data can be either

TABLE I
COMPARISON BETWEEN OUR SURVEY AND OTHER RELATED SURVEYS

| Survey | Year | Domain | | Scope | | | |
|---|---|---|---|---|---|---|---|
| | | Specific | General | Forecasting | Classification | Anomaly Detection | Imputation |
| Wang et al. [11] | 2019 | | ✔ | ● | ● | ● | ○ |
| Ye et al. [12] | 2020 | ✔ | | ● | ◐ | ○ | ○ |
| Jiang and Luo [13] | 2021 | ✔ | | ● | ○ | ○ | ○ |
| Bui et al. [14] | 2021 | ✔ | | ● | ○ | ○ | ○ |
| Jin et al. [15] | 2023 | ✔ | | ● | ○ | ○ | ○ |
| Al Sahili and Awad [16] | 2023 | | ✔ | ◐ | ○ | ○ | ○ |
| Rahmani et al. [17] | 2023 | | ✔ | ● | ○ | ○ | ◐ |
| **Our Survey** | 2024 | | ✔ | ● | ● | ● | ● |

* Specifically, ○ represents "Not Covered", ◐ signifies "Partially Covered", and ● corresponds to "Fully Covered".

*regularly* or *irregularly sampled*, with the latter also referred to as time series data with missing values. Within each of these cases, the data can be further classified into two primary types: *univariate* and *multivariate time series*. In the sequel, we employ bold uppercase letters (e.g., $\mathbf{X}$), bold lowercase letters (e.g., $\mathbf{x}$), and calligraphic letters (e.g., $\mathcal{V}$) to denote matrices, vectors, and sets, respectively.

*Definition 1 (Univariate Time Series):* A univariate time series is a sequence of scalar observations collected over time, which can be regularly or irregularly sampled. A regularly sampled univariate time series is defined as $\mathbf{X} = \{x_1, x_2, \ldots, x_T\} \in \mathbb{R}^T$, where $x_t \in \mathbb{R}$. For an irregularly sampled univariate time series, observations are collected at non-uniform time intervals, such as $\mathbf{X} = \{(t_1, x_1), (t_2, x_2), \ldots, (t_T, x_T)\} \in \mathbb{R}^T$, where time points are non-uniformly spaced.

*Definition 2 (Multivariate Time Series):* A multivariate time series is a sequence of $N$-dimensional vector observations collected over time, i.e., $\mathbf{X} \in \mathbb{R}^{N \times T}$. A regularly sampled multivariate time series has vector observations collected at uniform time intervals, i.e., $\mathbf{x}_t \in \mathbb{R}^N$. In an irregularly sampled multivariate time series, there are possibly $N$ unaligned time series with respect to time steps, which implies only $0 \leq n \leq N$ observations available at each time step.

The majority of research based on GNNs focuses on modeling multivariate time series, as they can be naturally abstracted into *spatial-temporal graphs*. This abstraction allows for an accurate characterization of dynamic inter-temporal and inter-variable dependencies. The former describes the relations between different time steps within each time series (e.g., the temporal dynamics of red nodes between $t_1$ and $t_3$ in Fig. 2), while the latter captures dependencies between time series (e.g., the spatial relations between four nodes at each time step in Fig. 2), such as the geographical information of the sensors generating the data for each variable. To illustrate this, we first define *attributed graphs*.

*Definition 3 (Attributed Graph):* An attributed graph is a static graph that associates each node with a set of attributes, representing node features. Formally, an attributed graph is defined as $\mathcal{G} = (\mathbf{A}, \mathbf{X})$, which consists of a (weighted) adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ and a node-feature matrix $\mathbf{X} \in \mathbb{R}^{N \times D}$. The adjacency matrix represents the graph topology, which can be

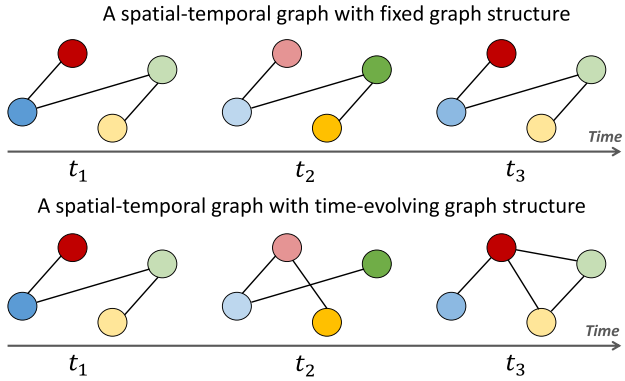



Fig. 2. Examples of spatial-temporal graphs, where node colors represent distinct features. The top and bottom panels demonstrate spatio-temporal graphs with fixed and dynamic graph structures over time, respectively.

characterized by $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$, the set of $N$ nodes, and $\mathcal{E} = \{e_{ij} := (v_i, v_j) \in \mathcal{V} \times \mathcal{V} \mid \mathbf{A}_{ij} \neq 0\}$, the set of edges; $\mathbf{A}_{ij}$ is the $(i, j)$-th entry in the adjacency matrix $\mathbf{A}$. The feature matrix $\mathbf{X}$ contains the node attributes, where the $i$-th row $\mathbf{x}_i \in \mathbb{R}^D$ represents the $D$-dimensional feature vector of node $v_i$.

In attributed graphs, multi-dimensional edge features can be considered too, however, this paper assumes only scalar weights encoded in the adjacency matrix to avoid overwhelming notations.

In light of this, a spatial-temporal graph can be described as a series of attributed graphs, which effectively represent (multivariate) time series data in conjunction with either evolving or fixed structural information over time.

*Definition 4 (Spatial-Temporal Graph):* A spatial-temporal graph can be interpreted as a *discrete-time dynamic graph* [31], [39], i.e., $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_T\}$, where $\mathcal{G}_t = (\mathbf{A}_t, \mathbf{X}_t)$ denotes an attributed graph at time $t$. $\mathbf{A}_t \in \mathbb{R}^{N \times N}$ and $\mathbf{X}_t \in \mathbb{R}^{N \times D}$ are corresponding adjacency and feature matrices. $\mathbf{A}_t$ may either evolve over time or remain fixed, depending on specific settings. When abstracting time series data, we let $\mathbf{X}_t := \mathbf{x}_t \in \mathbb{R}^N$.

We introduce graph neural networks as modern deep learning models to process graph-structured data. The core operation in typical GNNs, often referred to as *graph convolution*, involves exchanging information across neighboring nodes. In the context of time series analysis, this operation enables us to explicitly

rely on the inter-variable dependencies represented by the graph edges. Aware of the different nuances, we define GNNs in the spatial domain, which involves transforming the input signal with learnable functions along the dimension of $N$.

*Definition 5 (Graph Neural Network):* Given an attributed graph $\mathcal{G} = (\mathbf{A}, \mathbf{X})$, we define $\mathbf{x}_i = \mathbf{X}[i, :] \in \mathbb{R}^D$ as the $D$-dimensional feature vector of node $v_i$. A GNN learns node representations through two primary functions [40]: AGGREGATE$(\cdot)$ and COMBINE$(\cdot)$. The AGGREGATE$(\cdot)$ function computes and aggregates messages from neighboring nodes, while the COMBINE$(\cdot)$ function merges the aggregated and previous states to transform node embeddings. Formally, the $k$-th layer in a GNN is defined by the extended

$$\mathbf{a_i^{(k)}} = \text{AGGREGATE}^{(k)}\left(\left\{\mathbf{h_j^{(k-1)}} : v_j \in \mathcal{N}(v_i)\right\}\right),$$
$$\mathbf{h_i^{(k)}} = \text{COMBINE}^{(k)}\left(\mathbf{h_i^{(k-1)}}, \mathbf{a_i^{(k)}}\right), \tag{1}$$

or, more generally, aggregating messages computed from both sending and receiving nodes $v_j$ and $v_i$, respectively. Here, $\mathbf{a_i^{(k)}}$ and $\mathbf{h_i^{(k)}}$ represent the aggregated message from neighbors and the transformed node embedding of node $v_i$ in the $k$-th layer, respectively. The input and output of a GNN are $\mathbf{h_i^{(0)}} := \mathbf{x}_i$ and $\mathbf{h_i^{(K)}} := \mathbf{h_i}$.

The above formulation in (1) is referred to as *spatial GNNs*, as opposed to *spectral GNNs* which defines convolution from the lens of spectral graph theory. We refer the reader to recent publication [28] for a deeper analysis of spectral versus spatial GNNs, and [29] for a comprehensive review of GNNs.

To employ GNNs for time series analysis, it is implied that a graph structure must be provided. However, not all time series data have readily available graph structures and, in practice, two types of strategies are utilized to generate the missing graph structures from the data: *heuristics* or *learned* from data.

*Heuristic-Based Graphs:* This group of methods extracts graph structures from data based on heuristics, such as:

- *Spatial Proximity:* This approach defines the graph structure by considering the proximity between pairs of nodes based on, e.g., their geographical location. A typical example is the construction of the adjacency matrix $\mathbf{A}$ based on the shortest travel distance between nodes when the time series data have geospatial properties:

$$\mathbf{A}_{i,j} = \begin{cases} \frac{1}{dij}, & \text{if } d_{ij} \neq 0, \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

  where $d_{ij}$ denotes the shortest travel distance between node $i$ and node $j$. Some common kernel functions, e.g., Gaussian radial basis, can also be applied [15].
- *Pairwise Connectivity:* In this approach, the graph structure is determined by the connectivity between pairs of nodes, like that determined by transportation networks. The adjacency matrix $\mathbf{A}$ is defined as:

$$\mathbf{A}_{i,j} = \begin{cases} 1, & \text{if } v_i \text{ and } v_j \text{ are directly linked}, \\ 0, & \text{otherwise}. \end{cases} \tag{3}$$

  Typical scenarios include edges representing roads, railways, or adjacent regions [41], [42]. In such cases, the graph can be undirected or directed, resulting in symmetric and asymmetric adjacency matrices.
- *Pairwise Similarity:* This method constructs the graph by connecting nodes with similar attributes. A simple example is the construction of adjacency matrix $\mathbf{A}$ based on the cosine similarity between time series:

$$\mathbf{A}_{i,j} = \frac{\mathbf{x}_i^\top \mathbf{x}_j}{\|\mathbf{x}_i\| \|\mathbf{x}_j\|}, \tag{4}$$

  where $\|\cdot\|$ denotes the euclidean norm. There are also several variants for creating similarity-based graphs, such as Pearson correlation coefficient (PCC) [43] and dynamic time warping (DTW) [44].
- *Functional Dependence:* This approach defines the graph structure based on known functional dependencies between pairs of nodes, such as direct causal relationships or dependence from common hidden factors. For instance, adjacency matrix $\mathbf{A}$ can be constructed based on Granger causality [45] as

$$\mathbf{A}_{i,j} = \begin{cases} 1, & \text{if node } j \text{ Granger-causes} \\ & \text{node } i \text{ at a significance level } \alpha, \\ 0, & \text{otherwise}. \end{cases} \tag{5}$$

  Other examples involve transfer entropy (TE) [46] and directed phase lag index (DPLI) [47]. While overlap with previous heuristics exists, functional relations typically represent or estimate actual node dependencies in the data-generating process.

*Learning-Based Graphs:* In contrast to heuristic-based methods, learning-based approaches aim to learn the graph structure directly from the data and end-to-end with the downstream task. Accordingly, graph $\mathbf{A}$ can be defined as a function $\rho(\cdot)$ of some trainable model parameters $\Theta$ and, possibly, also of the time series observations $\mathbf{X}$, i.e., $\mathbf{A} = \rho(\Theta, \mathbf{X})$. Embedding-based approaches (e.g., [48], [49]) define the presence of each edge by comparing learned embedding vectors of the associated nodes; as an example, consider $\mathbf{A}_{i,j} = \text{ReLU}(\Theta_i^\top \Theta_j)$ with $\Theta_i, \Theta_j$ node embedding of nodes $i, j$, respectively. Another common example involves defining $\mathbf{A}$ through attention scores computed between node signals [50]. Sparsification methods – such as the ReLU activation above or top-k selection – are often applied to discard edges with null or low weights, thereby reducing the computational load requested by dense GNN operations [51]. An alternative strategy is modeling $\mathbf{A}$ as a discrete random variable whose parametric distribution, say $p_\Theta(\mathbf{A} \mid \mathbf{X})$, is learned alongside the other model parameters [52], [53]. Learning-based approaches enable the data-driven discovery of less obvious graph structures that are tailored to solve the given task, potentially providing more effective relations than heuristic-based graphs.

## III. FRAMEWORK AND CATEGORIZATION

In this section, we present a comprehensive task-oriented taxonomy for GNNs within the context of time series analysis (Section III-A). Subsequently, we investigate how to encode time series across various tasks by introducing a unified methodological framework for GNN architectures (Section III-B). According


Fig. 3. Task-oriented taxonomy of graph neural networks for time series analysis in the existing literature.

to the framework, all architectures are composed of a similar graph-based processing module $f_\theta$ and a second module $p_\phi$ specialized in downstream tasks.

### A. Task-Oriented Taxonomy

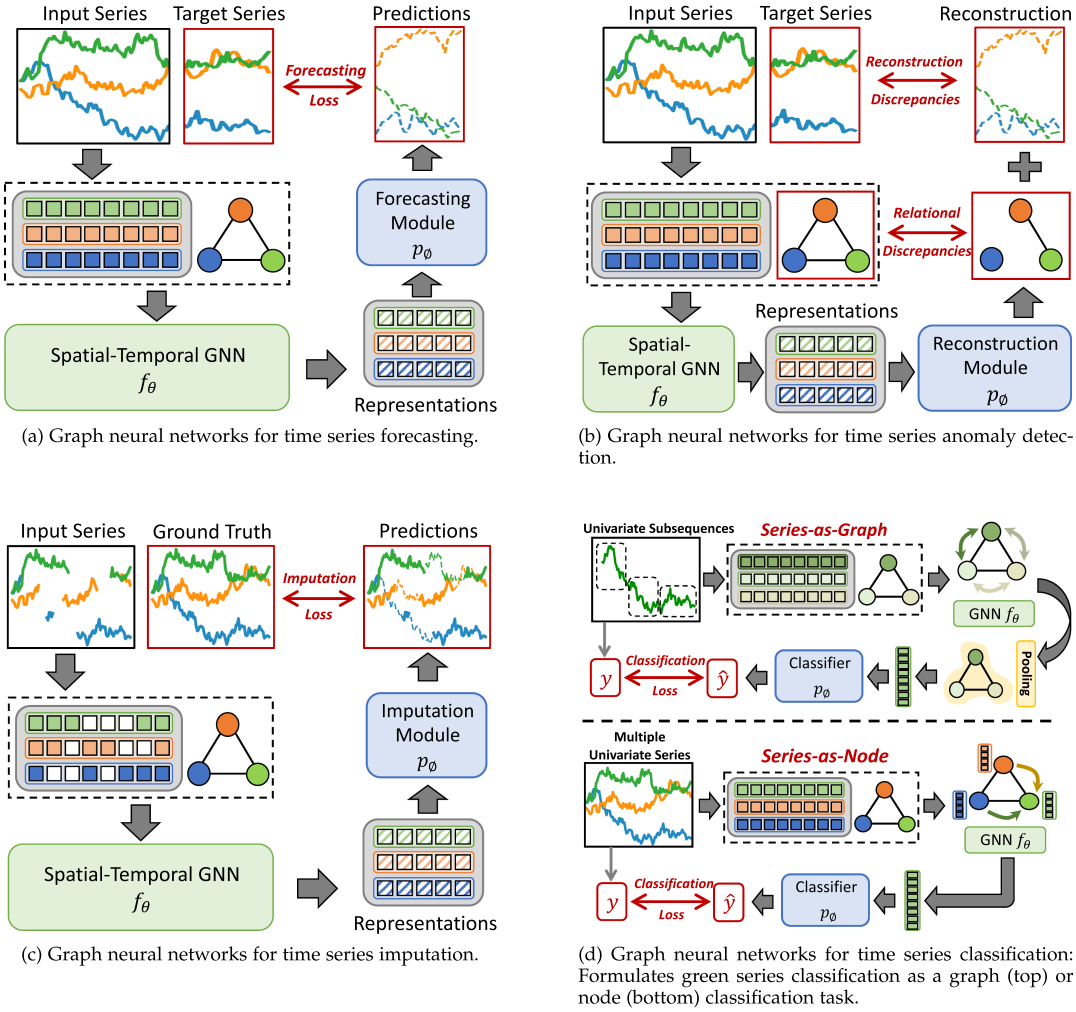
In Fig. 3, we illustrate a task-oriented taxonomy of GNNs encompassing the primary tasks and mainstream modeling for time series analysis, and showcasing the potential of GNN4TS. This survey focuses on four categories: time series *forecasting*, *anomaly detection*, *imputation*, and *classification*. These tasks are performed on top of the time series representations learned by *spatial-temporal graph neural networks* (STGNNs), which serve as the foundation for encoding time series data in existing literature across various tasks. We detail this in Section III-B.

*Time Series Forecasting:* This task is centered around predicting future values of the time series based on historical observations, as depicted in Fig. 4(a). Depending on application needs, we categorize this task into two types: *single-step-ahead forecasting* and *multi-step-ahead forecasting*. The former is meant to predict single future observations of the time series once at a time, i.e., the target at time $t$ is $\mathbf{Y} := \mathbf{X}_{t+H}$ for some $H \in \mathbb{N}$ steps ahead, while the latter makes predictions for a time interval, e.g., $\mathbf{Y} := \mathbf{X}_{t+1:t+H}$. Parameterized solutions to both predictive cases can be derived by optimizing

$$\theta^*, \phi^* = \arg\min_{\theta,\phi} \mathcal{L}_{\text{F}} \left( p_\phi \left( f_\theta(\mathbf{X}_{t-T:t}, \mathbf{A}_{t-T:t}) \right), \mathbf{Y} \right), \quad (6)$$

where $f_\theta(\cdot)$ and $p_\phi(\cdot)$ represent a spatial-temporal GNN and the predictor, respectively. Details regarding the $f_\theta(\cdot)$ architecture are given in Section III-B while the predictor is, normally, a multi-layer perceptron. In the sequel, we denote by $\mathbf{X}_{t-T:t}$ and $\mathbf{A}_{t-T:t}$ a spatial-temporal graph $\mathcal{G} = \{\mathcal{G}_{t-T}, \mathcal{G}_{t-T+1}, \ldots, \mathcal{G}_t\}$ with length $T$. If the underlying graph structure is fixed, then $\mathbf{A}_t := \mathbf{A}$. $\mathcal{L}_{\text{F}}(\cdot)$ denotes the forecasting loss, which is typically a squared or absolute loss function, e.g., STGCN [54] and MTGNN [51]. Most existing works minimize the error between the forecasting and the ground truth $\mathbf{Y}$ through (6); this process is known as *deterministic* time series forecasting. Besides, we have *probabilistic* time series forecasting methods, such as DiffSTG [55], that share the same objective (6) function though it is not directly optimized. Based on the size of the forecasting horizon $H$, we end up in either *short-term* or *long-term forecasting*.

*Time Series Anomaly Detection:* This task focuses on detecting irregularities and unexpected events in time series data (Fig. 4(b)). Detecting anomalies requires determining *when* the anomalous event occurred, while *diagnosing* them requests gaining insights about how and why the anomaly occurred. Due




(a) Graph neural networks for time series forecasting.

(b) Graph neural networks for time series anomaly detection.

(c) Graph neural networks for time series imputation.

(d) Graph neural networks for time series classification: Formulates green series classification as a graph (top) or node (bottom) classification task.

Fig. 4. Four categories of graph neural networks for time series analysis. For the sake of simplicity and illustrative purposes, we assume the graph structures are fixed in all subplots.

to the general difficulty of acquiring anomaly events, current research commonly treats anomaly detection as an unsupervised problem that involves the design of a model describing normal, non-anomalous data. The learned model is then used to detect anomalies by generating a high score whenever an anomaly event occurs. This model learning process mirrors the forecasting optimization, (6), with $f_\theta(\cdot)$ and $p_\phi(\cdot)$ denote the spatial-temporal GNN and the predictor, respectively. In general, the spatial-temporal GNN and the predictor are trained on normal, non-anomalous data using either forecasting [36], [56] or reconstruction [35], [57] optimization approaches, with the aim of minimizing the discrepancy between the normal input and the forecast (or reconstructed) series. Nonetheless, when these models are put to use for detecting anomalies, they are expected to fail in minimizing this discrepancy upon receiving anomalous input. This inability to conform to the expected low-discrepancy model behavior during anomaly periods creates a detectable difference, facilitating the detection of anomalies. The threshold separating normal and anomalous data is a sensitive hyperparameter that should be set considering the rarity of anomalies and aligned with a desired false alarm rate [58]. Lastly, to diagnose the causes of anomalies, a common strategy involves calculating discrepancies for each channel node and consolidating these into a single anomaly score [59]. This approach allows for the identification of the channel variables responsible for the anomaly events by calculating their respective contributions to the final score.

*Time Series Imputation:* This task is centered around estimating and filling in missing or incomplete data points within a time series (Fig. 4(c)). Current research in this domain can be broadly classified into two main approaches: *in-sample imputation* and *out-of-sample imputation*. In-sample imputation involves filling missing values in a given time series, while out-of-sample imputation pertains to inferring missing data not present in the training dataset. We formulate the learning objective as follows:

$$\theta^*, \phi^* = \arg\min_{\theta,\phi} \mathcal{L}_1 \left( p_\phi \left( f_\theta(\tilde{\mathbf{X}}_{t-T:t}, \mathbf{A}_{t-T:t}) \right), \mathbf{X}_{t-T:t} \right), \tag{7}$$

where $f_\theta(\cdot)$ and $p_\phi(\cdot)$ denote the spatial-temporal GNN and imputation module to be learned, respectively. The imputation module can e.g., be a multi-layer perceptron. In this task, $\tilde{\mathbf{X}}_{t-T:t}$ represents input time series data with missing values (reference time series), while $\mathbf{X}_{t-T:t}$ denotes the same time series without

missing values. As it is impossible to access the reference time series during training, a surrogate optimization objective is considered, such as generating synthetic missing values [37]. In (7), $\mathcal{L}_{\mathrm{I}}(\cdot)$ refers to the imputation loss, which can be, for instance, an absolute or a squared error, similar to forecasting tasks. For in-sample imputation, the model is trained and evaluated on $\tilde{\mathbf{X}}_{t-T:t}$ and $\mathbf{X}_{t-T:t}$. Instead, for out-of-sample imputation, the model is trained and evaluated on disjoint sequences, e.g., trained on $\tilde{\mathbf{X}}_{t-T:t}$ but evaluated on $\mathbf{X}_{t:t+H}$, where the missing values in $\tilde{\mathbf{X}}_{t:t+H}$ will be estimated. Similar to time series forecasting and anomaly detection, the imputation process can be either *deterministic* or *probabilistic*. The former predicts the missing values directly (e.g., GRIN [37]), while the latter estimates the missing values from data distributions (e.g., PriSTI [38]).

*Time Series Classification:* This task aims to assign a categorical label to a given time series based on its underlying patterns or characteristics. Rather than capturing patterns within a time series data sample, the essence of time series classification resides in discerning differentiating patterns that help separate samples based on their class labels. The optimization problem can be expressed as:

$$\theta^*, \phi^* = \arg\min_{\theta,\phi} \mathcal{L}_{\mathrm{C}} \left( p_\phi \left( f_\theta(\mathbf{X}, \mathbf{A}) \right), \mathbf{Y} \right), \tag{8}$$

where $f_\theta(\cdot)$ and $p_\phi(\cdot)$ denote, e.g., a GNN and a classifier to be learned, respectively. Using univariate time series classification as an example Fig. 4(d), the task can be formulated as either a graph or node classification task. In the case of graph classification (*Series-as-Graph*) [96], each series is transformed into a graph, and the graph will be the input of a GNN to generate a classification output. This can be achieved by dividing a series into multiple subsequences with a window size, $W$, serving as graph nodes, $\mathbf{X} \in \mathbb{R}^{N \times W}$, and an adjacency matrix, $\mathbf{A}$, describing the relationships between subsequences. A simple GNN, $f_\theta(\cdot)$, then employs graph convolution and pooling to obtain a condensed graph feature to be exploited by a classifier $p_\phi(\cdot)$ which assigns a class label to the graph. Alternatively, the node classification formulation (*Series-as-Node*), treats each series as a node in a dataset graph. Series-as-Node constructs an adjacency matrix representing the relationships between multiple distinct series in a given dataset [63]. With several series of length $T$ stacked into a matrix $\mathbf{X} \in \mathbb{R}^{N \times T}$ as node features and $\mathbf{A}$ representing pairwise relationships, the GNN operation, $f_\theta(\cdot)$, aims at leveraging the relationships across different series for accurate node series classification [97]. In all cases, $\mathbf{Y}$ is typically a one-hot encoded vector representing the categorical label of a univariate or multivariate time series.

### B. Unified Methodological Framework

In Fig. 5, we present a unified methodological framework of STGNNs mentioned in Section III-A for time series analysis. Specifically, our framework serves as the basis for encoding time series data in the existing literature across various downstream tasks (Fig. 3). As an extension, STGNNs incorporate spatial information by considering the relationships between nodes in the graph and temporal information by taking into account the evolution of node attributes over time. Similar to [15], we systematically categorize STGNNs from three perspectives: *spatial module*, *temporal module*, and *overall model architecture*.

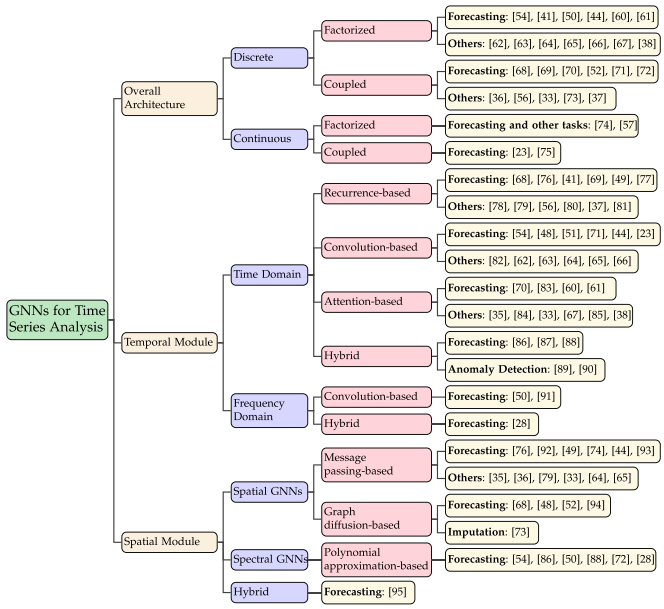



Fig. 5. Methodology-oriented taxonomy of graph neural networks for time series analysis.

*Spatial Module:* To model dependencies between time series over time, STGNNs employ the design principles of GNNs on static graphs. These can be further categorized into three types: *spectral GNNs*, *spatial GNNs*, and a combination of both (i.e., *hybrid*) [29]. Spectral GNNs are based on spectral graph theory and use the graph shift operator (like the graph Laplacian) to capture node relationships in the graph frequency domain [28], [98], [99]. Differently, spatial GNNs simplify spectral GNNs by directly designing filters that are localized to each node's neighborhood. The approaches in this category can be broadly classified into two types: *graph diffusion-based* and *message passing-based*. Notably, graph transformers [100] represent a specialized extension of message passing neural networks [101], further expanding the capabilities of this paradigm. Hybrid approaches combine both spectral and spatial methodologies to capitalize on the strengths of each method.

*Temporal Module:* To account for temporal dependencies in time series, STGNNs incorporate temporal modules that work in tandem with spatial modules to model intricate spatial-temporal patterns. Temporal dependencies can be represented in either the *time* or *frequency* domains. The approaches in the first category encompass *recurrence-based* (e.g., RNNs [26]), *convolution-based* (e.g., TCNs [102]), *attention-based* (e.g., transformers [27]), and a combination of these (i.e., *hybrid*). For the second category, analogous techniques are employed, followed by orthogonal space projections [28], such as the Fourier transform.

*Model Architecture:* To integrate the two modules, existing STGNNs are either *discrete* or *continuous* in terms of their overall neural architectures. Both types can be further subdivided into two subcategories: *factorized* and *coupled*. With typical

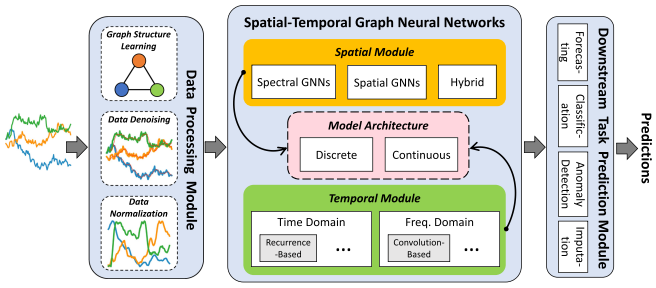



Fig. 6. General pipeline for time series analysis using graph neural networks.

factorized STGNN model architectures, the temporal processing is performed either before or after the spatial processing, whether in a discrete (e.g., STGCN [54]) or continuous manner (e.g., STGODE [74]). Conversely, the coupled model architecture refers to instances where spatial and temporal modules are interleaved, such as DCRNN [68] (discrete) and MTGODE [23] (continuous). Other authors refer to very related categories as time-then-space and time-and-space [103].

*General Pipeline:* In Fig. 6, we showcase a general pipeline that shows how STGNNs can be integrated into time series analysis. Given a time series dataset, we first process it using the *data processing module*, which performs essential data cleaning and normalization tasks, including the extraction of time series topology (i.e., graph structures). Subsequently, STGNNs are utilized to obtain time series representations, which can then be passed to different handlers (i.e., *downstream task prediction module*) to execute various analytical tasks, such as forecasting and anomaly detection.

## IV. GNNs for Time Series Forecasting

Time series forecasting aims to predict future time series values based on historical observations. While deep learning models have demonstrated considerable success in forecasting time series by capturing nonlinear temporal and spatial patterns more effectively than the linear counterpart [23], many of these approaches, such as LSTNet [111] and TPA-LSTM [112], overlook and implicitly model the rich underlying dynamic spatial correlations between time series. Recently, graph neural network (GNN)-based methods have shown great potential in explicitly and effectively modeling spatial and temporal dependencies in multivariate time series data, leading to enhanced forecasting performance.

GNN-based forecasting models can be categorized and examined from multiple perspectives. In terms of forecasting tasks, while many models focus on *multi-step forecasting* (i.e., predicting multiple consecutive steps ahead based on historical observations), a minority also discuss *single-step forecasting* (i.e., predicting the next or one arbitrary step ahead). From a methodological standpoint, these models can be dissected from three aspects: (1) modeling spatial (i.e., inter-variable) dependencies, (2) modeling inter-temporal dependencies, and (3) the fusion of spatial and temporal modules for time series forecasting. A summary of representative works is given in Table II.

### A. Modeling Inter-Variable Dependencies

Spatial dependencies, or inter-variable relationships, play a pivotal role in affecting a model's forecasting capability [28]. When presented with time series data and corresponding graph structures that delineate the strength of interconnections between time series, current studies typically employ (1) *spectral GNNs*, (2) *spatial GNNs*, or (3) a *hybrid* of both to model these spatial dependencies. At a high level, these methods all draw upon the principles of graph signal processing (as detailed in Definition 5 and subsequent discussion). Considering input variables $\mathbf{X}_t$ and $\mathbf{A}_t$ at a given time $t$, the goal here is to devise an effective GNN-based model $\textsc{Spatial}(\cdot)$ to adeptly capture salient patterns from different time series. This can be expressed as $\hat{\mathbf{X}}_t = \textsc{Spatial}(\mathbf{X}_t, \mathbf{A}_t)$, where $\hat{\mathbf{X}}_t$ collects all time series representations at time $t$ with spatial dependencies embedded.

*Spectral GNN-Based Approaches:* Early GNN-based forecasting models predominantly utilized ChebConv [113] to approximate graph convolution with Chebyshev polynomials, thereby modeling inter-variable dependencies. For instance, STGCN [54] intersects temporal convolution [114] and ChebConv layers to capture both spatial and temporal patterns. StemGNN [50] further proposes spectral-temporal graph neural networks to extract rich time series patterns by leveraging ChebConv and frequency-domain convolution neural networks. Other relevant research has largely followed suit, employing ChebConv to model spatial time series dependencies, while introducing novel modifications. These include attention mechanisms [86], [105], multi-graph construction [41], [72], and combinations of the two [88].

*Spatial GNN-Based Approaches:* Inspired by the recent success of spatial GNNs [29], another line of research has been modeling inter-variable dependencies using message passing [115] or graph diffusion [116]. From the graph perspective, these methods are certain simplifications compared to those based on spectral GNNs, where strong local homophilies are emphasised [28], [117]. Early methods such as DCRNN [68] and Graph WaveNet [48] incorporated graph diffusion layers into GRU [118] or temporal convolution to model time series data. In contrast, STGCN$(1^{st})$ (a second version of STGCN [54]) and ST-MetaNet [76] modeled spatial dependencies with GCN [119] and GAT [120] to aggregate information from adjacent time series. To enhance learning capabilities, several improvements have been proposed. For example, STSGCN [92] proposed spatial-temporal synchronous graph convolution, extending GCN to model spatial and temporal dependencies on localized spatial-temporal graphs. MTGODE [23] and TPGNN [60] proposed continuous graph propagation and graph propagation based on temporal polynomial coefficients. Additionally, recent approaches based on graph transformer [100] or hypergraphs [121], such as [122] and [110], can capture longer-range spatial dependencies due to their global receptive field, making them a separate branch of enhanced methods.

*Hybrid Approaches:* Some hybrid models also exist, integrating both spectral and spatial GNNs. For instance, SLCNN [95] employs ChebConv and localized message passing as global and local convolutions to capture spatial relations at multiple

TABLE II
SUMMARY OF REPRESENTATIVE GRAPH NEURAL NETWORKS FOR TIME SERIES FORECASTING

| Approach | Year | Venue | Task | Architecture | Spatial Module | Temporal Module | Missing Values | Input Graph | Learned Relations | Graph Heuristics |
|---|---|---|---|---|---|---|---|---|---|---|
| DCRNN [68] | 2018 | ICLR | M-S | D-C | Spatial GNN | T-R | No | R | - | SP |
| STGCN [54] | 2018 | IJCAI | M-S | D-F | Spectral GNN | T-C | No | R | - | SP |
| ST-MetaNet [76] | 2019 | KDD | M-S | D-F | Spatial GNN | T-R | No | R | - | SP, PC |
| NGAR [104] | 2019 | IEEE IJCNN | S-S | D-F | Spatial GNN | T-R | No | R | - | - |
| ASTGCN [86] | 2019 | AAAI | M-S | D-F | Spectral GNN | T-H | No | R | - | SP, PC |
| ST-MGCN [41] | 2019 | AAAI | S-S | D-F | Spectral GNN | T-R | No | R | - | SP, PC, PS |
| Graph WaveNet [48] | 2019 | IJCAI | M-S | D-F | Spatial GNN | T-C | No | O | S | SP |
| MRA-BGCN [69] | 2020 | AAAI | M-S | D-C | Spatial GNN | T-R | No | R | - | SP |
| MTGNN [51] | 2020 | KDD | S-S, M-S | D-F | Spatial GNN | T-C | No | NR | S | - |
| STGNN* [87] | 2020 | WWW | M-S | D-C | Spatial GNN | T-H | No | R | - | SP |
| GMAN [70] | 2020 | AAAI | M-S | D-C | Spatial GNN | T-A | No | R | - | SP |
| SLCNN [95] | 2020 | AAAI | M-S | D-F | Hybrid | T-C | No | NR | S | - |
| STSGCN [92] | 2020 | AAAI | M-S | D-C | Spatial GNN | T | No | R | - | PC |
| StemGNN [50] | 2020 | NeurIPS | M-S | D-F | Spectral GNN | F-C | No | NR | S | - |
| AGCRN [49] | 2020 | NeurIPS | M-S | D-C | Spatial GNN | T-R | No | NR | S | - |
| LSGCN [105] | 2020 | IJCAI | M-S | D-F | Spectral GNN | T-C | No | R | - | SP |
| STAR [83] | 2020 | ECCV | M-S | D-F | Spatial GNN | T-A | No | R | - | PC |
| GTS [52] | 2021 | ICLR | M-S | D-C | Spatial GNN | T-R | No | NR | S | - |
| GEN [106] | 2021 | ICLR | S-S | D-F | Spatial GNN | T-R | No | R | - | - |
| Z-GCNETs [71] | 2021 | ICML | M-S | D-C | Spatial GNN | T-C | No | NR | S | - |
| STGODE [74] | 2021 | KDD | M-S | C-F | Spatial GNN | T-C | No | R | - | SP, PS |
| STFGNN [44] | 2021 | AAAI | M-S | D-F | Spatial GNN | T-C | No | R | - | SP, PS |
| DSTAGNN [88] | 2022 | ICML | M-S | D-F | Spectral GNN | T-H | No | R | - | PC, PS |
| TPGNN [60] | 2022 | NeurIPS | S-S, M-S | D-F | Spatial GNN | T-A | No | NR | D | - |
| MTGODE [23] | 2022 | IEEE TKDE | S-S, M-S | C-C | Spatial GNN | T-C | No | NR | S | - |
| STG-NCDE [75] | 2022 | AAAI | M-S | C-C | Spatial GNN | T-C | Yes | NR | S | - |
| STEP [61] | 2022 | KDD | M-S | D-F | Spatial GNN | T-A | No | NR | S | - |
| Chauhan et al. [107] | 2022 | KDD | M-S | - | - | - | Yes | O | S | SP |
| RGSL [72] | 2022 | IJCAI | M-S | D-C | Spectral GNN | T-R | No | R | S | SP, PC |
| FOGS [108] | 2022 | IJCAI | M-S | - | - | - | No | NR | S | - |
| METRO [93] | 2022 | VLDB | M-S | D-C | Spatial GNN | T | No | NR | D | - |
| SGP [77] | 2023 | AAAI | M-S | D-F | Spatial GNN | T-R | No | R | - | SP, PS |
| HiGP [109] | 2023 | arXiv | M-S | D-F | Spatial GNN | T-R | No | R | S | SP, PS |
| Jin et al. [28] | 2023 | arXiv | M-S, M-L | D-F | Spectral GNN | F-H | No | NR | S | - |
| CaST [91] | 2023 | NeurIPS | M-S | D-F | Spectral GNN | T&F-C | No | O | S | PC |
| GPT-ST [110] | 2023 | NeurIPS | M-S | D-F | Spatial GNN | T-C | No | NR | S | - |

Task notation: The first letter, "M" or "S", indicates multi-step or single-step forecasting, and the second letter, "S" or "L", denotes short-term or long-term forecasting. Architecture notation: "D" and "C" represent "Discrete" and "Continuous"; "C" and "F" stand for "Coupled" and "Factorized". Temporal module notation: "T" and "F" signify "Time" and "Frequency" domains; "R", "C", "A", and "H" correspond to "Recurrence", "Convolution", "Attention", and "Hybrid". Input graph notation: "R" indicates that a pre-calculated graph structure (with a certain graph heuristic) is a required input of the model, "NR" that such graph is not required (not a model's input), while "O" signifies that the model can optionally exploit given input graphs. Notation of learned graph relations: "S" and "D" indicate "Static" and "Dynamic". Notation of adopted graph heuristics: "SP", "PC", "PS", and "FD" denote "Spatial Proximity", "Pairwise Connectivity", "Pairwise Similarity", and "Functional Dependency". The "Missing Values" column indicates whether corresponding methods can handle missing values in input time series.

TABLE III
SUMMARY OF REPRESENTATIVE GRAPH NEURAL NETWORKS FOR TIME SERIES ANOMALY DETECTION

| Approach | Year | Venue | Strategy | Spatial Module | Temporal Module | Missing Values | Input Graph | Learned Relations | Graph Heuristics |
|---|---|---|---|---|---|---|---|---|---|
| CCM-CDT [47] | 2019 | IEEE TNNLS | RC | Spatial GNN | T-R | No | R | - | PC, FD |
| MTAD-GAT [35] | 2020 | IEEE ICDM | FC+RC | Spatial GNN | T-A | No | NR | - | - |
| GDN [36] | 2021 | AAAI | FC | Spatial GNN | - | No | NR | S | - |
| GTA [89] | 2021 | IEEE IoT | FC | Spatial GNN | T-H | No | NR | S | - |
| EvoNet [78] | 2021 | WSDM | CL | Spatial GNN | T-R | No | R | - | PS |
| Event2Graph [84] | 2021 | arXiv | RL | Spatial GNN | T-A | No | R | - | PS |
| GANF [57] | 2022 | ICLR | RC+RL | Spatial GNN | T-R | No | NR | S | - |
| Grelen [90] | 2022 | IJCAI | RC+RL | Spatial GNN | T-H | No | NR | D | - |
| VGCRN [79] | 2022 | ICML | FC+RC | Spatial GNN | T-R | No | NR | S | - |
| FuSAGNet [56] | 2022 | KDD | FC+RC | Spatial GNN | T-R | No | NR | S | - |
| GTAD [130] | 2022 | Entropy | FC+RC | Spatial GNN | T-C | No | NR | - | - |
| HgAD [131] | 2022 | IEEE BigData | FC | Spatial GNN | - | No | NR | S | |
| HAD-MDGAT [80] | 2022 | IEEE Access | FC+RC | Spatial GNN | T-A | No | NR | - | - |
| STGAN [80] | 2022 | IEEE TNNLS | RC | Spatial GNN | T-R | No | R | - | SP |
| GIF [132] | 2022 | IEEE IJCNN | RC | Spatial GNN | - | No | R | - | SP, PC, FD |
| DyGraphAD [82] | 2023 | arXiv | FC+RL | Spatial GNN | T-C | No | R | - | PS |
| GraphSAD [133] | 2023 | arXiv | CL | Spatial GNN | T-C | No | R | - | PS, PC |
| CST-GL [62] | 2023 | IEEE TNNLS | FC | Spatial GNN | T-C | No | NR | S | - |

Strategy notation: "CL", "FC", "RC", and "RL" indicate "Class", "Forecast", "Reconstruction", and "Relational Discrepancies", respectively. The remaining notations are shared with Table II.

TABLE IV
SUMMARY OF GRAPH NEURAL NETWORKS FOR TIME SERIES CLASSIFICATION

| Approach | Year | Venue | Task | Conversion | Spatial Module | Temporal Module | Missing Values | Input Graph | Learned Relations | Graph Heuristics |
|---|---|---|---|---|---|---|---|---|---|---|
| MTPool [151] | 2021 | NN | M | - | Spatial GNN | T-C | No | NR | S | - |
| Time2Graph+ [96] | 2021 | IEEE TKDE | U | Series-as-Graph | Spatial GNN | - | No | R | - | PS |
| RainDrop [33] | 2022 | ICLR | M | - | Spatial GNN | T-A | Yes | NR | S | - |
| SimTSC [63] | 2022 | SDM | U+M | Series-as-Node | Spatial GNN | T-C | No | R | - | PS |
| LB-SimTSC [97] | 2023 | arXiv | U+M | Series-as-Node | Spatial GNN | T-C | No | R | - | PS |
| TodyNet [64] | 2023 | arXiv | M | - | Spatial GNN | T-C | No | NR | D | - |
| EC-GCN [152] | 2023 | Comput. Netw. | U | Series-as-Graph | Spatial GNN | T-C | No | R | D | PS |
| MTS2Graph [153] | 2024 | Pattern Recognit. | M | Series-as-Graph | Spatial GNN | T-C | No | NR | - | - |

Task notation: "U" and "M" refer to univariate and multivariate time series classification tasks. Conversion represents the transformation of a time series classification task into a graph-level task as either graph or node classification task, represented as "Series-as-Graph" and "Series-as-Node", respectively. The remaining notations are shared with Table II.

granularities. Conversely, Auto-STGNN [123] integrates neural architecture search to identify high-performance GNN-based forecasting models.

More details about modeling inter-variable dependencies in GNNs for time series forecasting are in *Appendix A.1*, available online.

### B. Modeling Inter-Temporal Dependencies

The modeling of temporal dependencies within time series represents another important element in various GNN-based forecasting methods. These dependencies (i.e., temporal patterns) are capable of being modeled in the *time* or/and *frequency* domains. A summary of representative methods, along with their temporal module classifications, is presented in Table II. Given a univariate time series $\mathbf{X}_n$ with length $T$, the primary goal here is to learn an effective temporal model, referred to as $\textsc{Temporal}(\cdot)$. This model is expected to accurately capture the dependencies between data points within $\mathbf{X}_n$, such that $\hat{\mathbf{X}}_n = \textsc{Temporal}(\mathbf{X}_n)$, where $\hat{\mathbf{X}}_n$ symbolizes the representation of time series $\mathbf{X}_n$. In the construction of $\textsc{Temporal}(\cdot)$, both the time and frequency domains can be exploited within *convolutional* and *attentive* mechanisms. *Recurrent* models can also be employed for modeling in the time domain specifically. Additionally, *hybrid* models exist in both domains, integrating different methodologies such as attention and convolution neural networks.

*Recurrent Models:* Several early methodologies rely on recurrent models for understanding inter-temporal dependencies in the time domain. For instance, DCRNN [68] integrates graph diffusion with gated recurrent units (GRU) [118] to model the spatial-temporal dependencies in traffic data. On a different note, AGCRN [49] merges GRU with a factorized variant of GCN [119] and a graph structure learning module. More recent studies, such as GTS [52] and RGSL [72], share similar designs but primarily emphasize different graph structure learning mechanisms.

*Convolution Models:* Convolutional neural networks (CNNs), on the other hand, provide a more efficient perspective for modeling inter-temporal dependencies, with the bulk of existing studies in the time domain. An instance of this is STGCN [54], which introduces temporal gated convolution that integrates 1-D convolution with gated linear units (GLU) to facilitate tractable model training. Other representative examples include MTGNN [51] and MTGODE [23]. An alternative strand of methodologies, including StemGNN [50] and TGC [28], focuses on modeling temporal clues in the frequency domain.

*Attention Models:* Recently, a growing number of methodologies is turning towards attention mechanisms, e.g., the self-attention in transformer [124], to embed temporal correlations. For instance, GMAN [70] attentively aggregates historical information by considering both spatial and temporal features. ST-GRAT [125] mirrors the transformer's architecture to embed historical observations in conjunction with its proposed spatial attention mechanism. Recent strides such as STEP [61] similarly employ transformer layers to model the temporal dependencies within each univariate time series.

*Hybrid Models:* Hybrid models also find application in modeling inter-temporal dependencies. For example, AST-GCN [86] and DSTAGNN [88] concurrently employ temporal attention and convolution in learning temporal correlations. STGNN* [87] amalgamates both GRU and transformer to capture local and global temporal dependencies. In the frequency domain, the nonlinear variant of TGC [28] captures temporal relations through the combination of spectral attention and convolution models.

Complete discussion of modeling inter-temporal dependencies is in *Appendix A.2*, available online.

### C. Forecasting Architectural Fusion

Given the spatial and temporal modules discussed, denoted as $\textsc{Spatial}(\cdot)$ and $\textsc{Temporal}(\cdot)$, four categories of neural architectural fusion have been identified as effective means to capture spatial-temporal dependencies within time series data: (1) *discrete factorized*, (2) *discrete coupled*, (3) *continuous factorized*, and (4) *continuous coupled*. In discrete factorized models, spatial and temporal dependencies are usually learned and processed independently. Discrete coupled models, on the other hand, explicitly or implicitly incorporate spatial and temporal modules into a singular process when modeling spatial-temporal dependencies. Different from discrete models, some methods abstract the underlying modeling processes with neural differential equations, which we categorize as continuous models. Specifically, continuous factorized models involve distinct processes, either partially or entirely continuous (e.g., [74]), to model spatial and temporal dependencies. In contrast, continuous coupled models employ a single continuous process to accomplish this task, such as [23] and [75].

*Discrete Architectures:* Numerous existing GNN-based time series forecasting methods are models processing discrete data. For instance, factorized approaches like STGCN [54] employ a sandwich structure of graph and temporal gated convolution layers as its fundamental building block. Subsequent works, such as DGCNN [126] and HGCN [127], retain this model architecture while introducing enhancements such as dynamic graph structure estimation and hierarchical graph generation. In the realm of discrete coupled models, early works such as DCRNN [68] and Cirstea et al. [94] straightforwardly incorporate graph diffusion or attention models into recurrent units. Subsequent works, such as MRA-BGCN [69] and RGSL [72], are based on similar concepts but with varying implementations. There are also some studies integrating spatial and temporal convolution or attention operations into a single module. An example is GMAN [70], which proposes a building block that integrates the spatial and temporal attention mechanisms in a gated manner.

*Continuous Architectures:* To date, only a handful of methods fall into this category. For factorized methods, STGODE [74] proposes to depict the graph propagation as a continuous process with a neural ordinary differential equation (NODE) [128]. This approach allows for the effective characterization of long-range spatial-temporal dependencies in conjunction with dilated convolutions along the time axis. For coupled methods, MT-GODE [23] generalizes both spatial and temporal modeling processes found in most related works into a single unified process that integrates two NODEs. STG-NCDE [75] shares a similar idea but operates under the framework of neural controlled differential equations (NCDEs) [129].

Refer to *Appendix A.3*, available online, for more details about the architecture fusion in GNNs for time series forecasting.

## V. GNNs for Time Series Anomaly Detection

Time series anomaly detection identifies data observations that deviate from the nominal data-generating process [134]. Anomalies, defined as such deviations, contrast with normal data. Terms like novelty and outlier are often used interchangeably with anomaly [135]. Deviations can be single observations (points) or a series of observations (subsequences) [136]. However, unlike normal time series data, anomalies are hard to characterize because they are rare, making them difficult to collect and label, and it is usually impossible to establish all potential anomalies, limiting supervised learning. Therefore, unsupervised detection techniques are widely explored as practical solutions.

Traditionally, distance-based [137] and distributional techniques [138] have been widely used for detecting irregularities in time series data. Recently, deep learning has driven significant advancements, particularly with recurrent models employing reconstruction [139] and forecasting [140] strategies. These models use forecast and reconstruction errors to measure discrepancies between expected and actual signals, as a model trained on normal data is more likely to fail with anomalous data. However, recurrent models [141] often lack explicit modeling of pairwise interdependence among variables, limiting their effectiveness in detecting complex anomalies [35], [142]. Recently, GNNs have shown promise in capturing temporal and spatial dependencies among variables, addressing this gap [36], [56].

### A. General Approach to Anomaly Detection

Treating anomaly detection as an unsupervised task relies on models to learn a general concept of what normality is for a given dataset [143], [144]. To achieve this, deep learning architectures deploy a bifurcated modular framework, constituted by a backbone module and a scoring module [145]. First, a backbone model, $\textsc{Backbone}(\cdot)$, is trained to fit given training data, assumed to be nominal, or to contain very few anomalies. Then, a scoring module, $\textsc{Scorer}(\mathbf{X}, \hat{\mathbf{X}})$, produces a score used to identify the presence of anomalies by comparing the output $\hat{\mathbf{X}} = \textsc{Backbone}(\mathbf{X})$ of the backbone module with the observed time series data $\mathbf{X}$. The score is intended as a measure of the discrepancy between the expected signals under normal and anomalous circumstances. Furthermore, it is also important for a model to diagnose anomaly events by pinpointing the responsible variables. Consequently, a scoring function typically computes the discrepancy for each individual channel first, before consolidating these discrepancies across all channels into a single anomaly value.

To provide a simple illustration of the entire process, the backbone can be a GNN forecaster that makes a one-step-ahead forecast for the scorer. The scorer then computes the anomaly score as the sum of the absolute forecast error for each channel variable, represented as $\sum_i^N |x_t^i - \hat{x}_t^i|$ across $N$ channel variables. Since the final score is computed based on the summation of channel errors, an operator can determine the root cause variables by computing the contribution of each variable to the summed error.

Advancements in the anomaly detection and diagnosis field have led to the proposal of more comprehensive backbone and scoring modules [136], [145], primarily driven by the adoption of GNN methodologies [35], [36], [56].

### B. Discrepancy Frameworks for Anomaly Detection

Most of anomaly detection models follow the same backbone-scorer architecture. However, the way the backbone module is trained to learn data structure from nominal data and the implementation of the scoring module differentiate these methods into three categories: (1) *reconstruction*, (2) *forecast*, and (3) *relational discrepancy* frameworks. A summary of representative works is provided in Table III.

*Reconstruction Discrepancy:* Reconstruction discrepancy frameworks operate on the assumption that reconstruction error is low during normal periods and high during anomalies. They are designed to replicate their inputs as outputs, similar to autoencoders [146]. The backbone is expected to effectively model and reconstruct the training data distribution but not out-of-sample data. To achieve this, these frameworks often include constraints and regularization, such as enforcing a low-dimensional embedded code [147] or using variational objectives [148]. Once the data structure is learned, the model should approximate the input well during normal periods but

struggle during anomalies. The SCORER$(\cdot)$ then computes a discrepancy score from the reconstructed outputs to identify anomalous events. Although deep reconstruction models generally follow these principles for detecting anomalies, a key distinction between GNNs and other architectural types rests in the backbone reconstructor, BACKBONE$(\cdot)$, which is characterized by its STGNN implementation. For example, MTAD-GAT [35] and LSTM-VAE [139] both use a variational objective [148], but MTAD-GAT employs a graph attention network as a spatial-temporal encoder to learn inter-variable and inter-temporal dependencies. Other similar works include VGCRN [79] and FuSAGNet [56]. Another research direction in this category of methods focused on graph-level embeddings to represent the input graph data as vectors to enable the application of well-established and sophisticated detection methods designed for multivariate time series [58], [149].

*Forecast Discrepancy:* Forecast discrepancy frameworks rely on the assumption that forecast error should be low during normal periods, but high during anomalous periods. Here, the backbone module is substituted with a GNN forecaster that is trained to predict a one-step-ahead forecast. During deployment, the forecaster makes a one-step-ahead prediction, and the forecasts are given to the scorer. The scorer compares the forecasts against the real observed signals to compute discrepancies such as absolute error [36] or mean-squared error [89]. Importantly, it is generally assumed that a forecasting-based model will exhibit erratic behavior during anomaly periods when the input data deviates from the normal patterns, resulting in a significant forecasting discrepancy. GDN [36] is a representative work, consisting of a graph structure module that learns the underlying topology and a graph attention network that encodes input series representations for one-step-ahead forecasts. Then, its scorer computes the forecast discrepancy as the maximum absolute forecast error among the channel variables to indicate whether an anomaly event has occurred. Other similar approaches include GTA [89] and CST-GL [62].

While forecast discrepancy frameworks can be similar to reconstruction discrepancy frameworks, they rely on fundamentally different principles and implementation strategies. Reconstruction discrepancy frameworks project current input onto a latent space, attempting to reconstruct it and identifying anomalies when reconstruction fails. This method uses current input data during both training and inference, focusing on the model's ability to reconstruct training data but not out-of-sample data. Conversely, forecast discrepancy frameworks use historical data to predict current values and identify anomalies by comparing predictions with actual observations. Reconstruction focuses on replicating current inputs, while forecasting emphasizes predicting future data points and detecting anomalies through prediction errors. Advanced forecast discrepancy-based methods, such as GST-Pro [150], can even predict anomalies without using actual observations, allowing anomaly prediction at future timestamps.

*Relational Discrepancy:* Relational discrepancy frameworks rely on the assumption that the relationship between variables should exhibit significant shifts from normal to anomalous periods. The logical evolution of using STGNN involves leveraging its capability to learn graph structures for both anomaly detection and diagnosis. In this context, the backbone serves as a graph learning module that constructs the hidden evolving relationship between variables. The scorer, on the other hand, is a function that evaluates changes in these relationships and assigns an anomaly or discrepancy score accordingly. GReLeN [90] pioneered the use of learned dynamic graphs to detect anomalies through relational discrepancies. Its reconstruction module dynamically constructs graph structures based on input time series data at each time point. These structures are then used by a scorer to compute changes in the in-degree and out-degree values of channel nodes. In contrast, DyGraphAD employs a forecasting approach [82]. It divides a multivariate series into subsequences, converts these into dynamically evolving graphs, and trains the network to predict one-step-ahead graph structures. The scorer in DyGraphAD computes the forecast error in the graph structure as the relational discrepancy for anomaly detection.

*Hybrid and Other Discrepancies:* Different discrepancy-based frameworks offer unique advantages for detecting various types of anomalies. As demonstrated in GDN [36], the relational discrepancy framework can uncover spatial anomalies that are concealed within the relational patterns between different channels. In contrast, forecast discrepancy frameworks excel at identifying temporal anomalies like sudden spikes or seasonal inconsistencies. A comprehensive solution would leverage the strengths of STGNNs by combining multiple discrepancy measures for anomaly detection. For example, MTAD-GAT [35] uses both reconstruction and forecast discrepancies, while DyGraphAD [82] combines forecast and relational discrepancies. Additionally, incorporating prior knowledge about anomalous behaviors can enhance detection. For instance, GraphSAD [133] creates pseudo labels for six distinct anomaly types on training data, transforming unsupervised anomaly detection into a standard classification task, with class discrepancy as the anomaly indicator.

Our detailed discussion about the GNNs for time series anomaly detection can be found in *Appendix B*, available online.

## VI. GNNs for Time Series Classification

Time series classification seeks to assign a categorical label to a given time series based on its underlying patterns or characteristics. As outlined in a recent survey [154], early literature primarily focused on distance-based approaches [155], [156] and ensembling methods [157], [158]. However, despite their strong performance, both strategies struggle with scalability for high-dimensional or large datasets [159], [160]. To address these limitations, researchers are exploring deep learning techniques to enhance the performance and scalability of time series classification. A comprehensive discussion can be found in the latest survey by Foumani et al. [154], though it does not cover the application of GNNs in this field. By transforming time series data into graph representations, one can leverage the powerful capabilities of GNNs to capture both local and global patterns. Furthermore, GNNs are capable of mapping the intricate relationships among different time series data samples within a particular dataset. In the following discussion, we provide a fresh

GNN perspective on the univariate and multivariate time series classification problem. A compilation of representative works is presented in Table IV

### A. Univariate Time Series Classification

Time series classification inherently differs from other time series analyses by focusing on discerning patterns that distinguish samples based on class labels, rather than capturing patterns within the data. Unlike forecasting future points or detecting real-time anomalies, it aims to identify divergent patterns *across series*. We explore two novel graph-based approaches for univariate time series classification: *Series-as-Graph* and *Series-as-Node*.

*Series-as-Graph:* This approach transforms a univariate time series into a graph to identify unique patterns that enable accurate classification using a GNN. First, each series is broken down into subsequences as nodes, and the nodes are connected with edges illustrating their relationships. Following this, a GNN is applied to make graph classification. This procedure is represented in the upper block of Fig. 4(d). Series-as-Graph was first presented in Time2Graph [161], which was later developed further with the incorporation of GNN, as Time2Graph+ [96]. The Time2Graph+ modeling process can be described as a two-step process: first, a time series is transformed into a shapelet graph, and second, a GNN is utilized to model the relations between shapelets along with a graph pooling operation to derive the global representation of the time series. This representation is then fed into a classifier to assign class labels to the time series. Similarly, EC-GCN [152] constructs graphs from encrypted traffic for classification. The Series-as-Graph approach can also be extended to multivariate time-series classification task [153]. While we use Series-as-Graph to describe the formulation of a time series classification task as a graph classification task, this should not be confused with Series2Graph [162], which is for anomaly detection tasks.

*Series-as-Node:* As capturing differentiating class patterns across different series data samples is important, leveraging relationships across the different series data samples in a given dataset can be beneficial for classifying a time series. To achieve this, one can take the Series-as-Node approach, where each series sample is seen as a separate node. These series nodes are connected with edges that represent the relationships between them, creating a large graph that provides a complete view of the entire dataset. This procedure is depicted in the lower block of Fig. 4(d), which essentially formulates a time series classification task as a node classification task. Series-as-Node was originally presented in SimTSC [63]. In this work, series nodes are connected using edges, which are defined by their pairwise DTW distance, to construct a graph. LB-SimTSC [97] further extends on SimTSC to improve the DTW preprocessing efficiency by employing the widely-used lower bound for DTW [163]. This allows for a time complexity of $O(L)$ rather than $O(L^2)$, dramatically reducing computation time.

More details about GNN-based univariate time series classification are in *Appendix C.1*, available online.

### B. Multivariate Time Series Classification

In essence, multivariate time series classification maintains fundamental similarities with its univariate counterpart. However, it introduces an additional layer of complexity: the necessity to capture intricate inter-variable dependencies. For example, given the interconnectedness of brain regions, analyzing a single node in isolation may not fully capture the comprehensive neural dynamics [164]. By modeling inter-variable dependencies, we can understand the relationships between different nodes, thereby offering a more holistic view of brain activity. This facilitates the differentiation of intricate patterns that can classify patients with and without specific neurological conditions. Since the relationships between the variables, or inter-variable dependencies, can be naturally thought of as a network graph, GNNs are ideally suitable as illustrated before in Section IV. The primary aim here is to effectively distill the complexity of high-dimensional series data into a more comprehensible, yet equally expressive, representation that enables differentiation of time series into their representative classes [33], [64], [151]. More discussion is in *Appendix C.2*, available online.

## VII. GNNs for Time Series Imputation

Time series imputation, a crucial task in numerous real-world applications, involves estimating missing values within one or more data point sequences. Traditional time series imputation approaches have relied on statistical methodologies, such as mean imputation, spline interpolation [171], and regression models [172]. However, these methods often struggle to capture complex temporal dependencies and non-linear relationships within the data. While some deep neural network-based works, such as [173] and [174], have mitigated these limitations, they do not explicitly consider inter-variable dependencies. The recent emergence of GNNs offers new possibilities for time series imputation by better capturing intricate spatial and temporal dependencies. From a task perspective, GNN-based time series imputation can be broadly categorized into two types: *in-sample imputation* and *out-of-sample imputation*. The former involves filling in missing values within the given time series data, while the latter predicts missing values in disjoint sequences [37]. From a methodological perspective, GNNs for time series imputation can be categorized into *deterministic* and *probabilistic* approaches. Deterministic imputation provides a single best estimate for missing values, while probabilistic imputation accounts for uncertainty by offering a distribution of possible values. In Table V, we summarize most of the related works on GNN for time series imputation to date, offering a comprehensive overview of the field and its current state of development.

### A. In-Sample Imputation

The majority of existing GNN-based methods primarily focus on in-sample time series data imputation. For instance, GACN [65] proposes to model spatial-temporal dependencies in time series data by interleaving GAT [120] and temporal convolution layers in its encoder. It then imputes the missing

TABLE V
SUMMARY OF GRAPH NEURAL NETWORKS FOR TIME SERIES IMPUTATION

| Approach | Year | Venue | Task | Type | Spatial Module | Temporal Module | Inductiveness | Input Graph | Learned Relations | Graph Heuristics |
|---|---|---|---|---|---|---|---|---|---|---|
| IGNNK [73] | 2021 | AAAI | Out-of-sample | Deterministic | Spatial GNN | - | Yes | R | - | SP, PC |
| GACN [65] | 2021 | ICANN | In-sample | Deterministic | Spatial GNN | T-C | No | R | - | PC |
| SATCN [66] | 2021 | arXiv | Out-of-sample | Deterministic | Spatial GNN | T-C | Yes | R | - | SP |
| GRIN [37] | 2022 | ICLR | Both | Deterministic | Spatial GNN | T-R | Yes | R | - | SP |
| SPIN [67] | 2022 | NIPS | In-sample | Deterministic | Spatial GNN | T-A | No | R | - | SP |
| FUNS [165] | 2022 | ICDMW | Out-of-sample | Deterministic | Spatial GNN | T-R | Yes | R | - | - |
| AGRN [166] | 2022 | ICONIP | In-sample | Deterministic | Spatial GNN | T-R | No | NR | S | - |
| MATCN [85] | 2022 | IEEE IoT-J | In-sample | Deterministic | Spatial GNN | T-A | No | R | - | - |
| PriSTI [38] | 2023 | arXiv | In-sample | Probabilistic | Spatial GNN | T-A | No | R | - | SP |
| DGCRIN [167] | 2023 | KBS | In-sample | Deterministic | Spatial GNN | T-R | No | NR | D | - |
| GARNN [168] | 2023 | Neurocomputing | In-sample | Deterministic | Spatial GNN | T-R | No | R | - | PC |
| MDGCN [81] | 2023 | Transp. Res. Part C | In-sample | Deterministic | Spatial GNN | T-R | No | R | - | SP, PS |
| INCREASE [169] | 2023 | WWW | Out-of-sample | Deterministic | Spatial GNN | T-R | Yes | R | - | SP, PC, FD |
| Yun et al. [170] | 2023 | OpenReview | In-sample | Probabilistic | Spatial GNN | - | No | R | - | - |

Task notation: "Out-of-sample", "In-sample", and "Both" refer to the types of imputation problems addressed by the approach. Type represents the imputation method as either deterministic or probabilistic. Inductiveness indicates if the method can generalize to unseen nodes. The remaining notations are shared with Table II.

data by combining GAT and temporal deconvolution layers that map latent states back to original feature spaces. GRIN [37] introduces the graph recurrent imputation network, where each unidirectional module consists of one spatial-temporal encoder and two different imputation executors. The spatial-temporal encoder adopted in this work combines MPNN [115] and GRU [118]. After generating the latent time series representations, the first-stage imputation fills missing values with one-step-ahead predicted values, which are then refined by a final one-layer MPNN before passing to the second-stage imputation for further processing. Similar works using bidirectional recurrent architectures include AGRN [166], DGCRIN [167], GARNN [168], and MDGCN [81], where the main differences lie in intermediate processes. Recently, a few research studies have explored probabilistic in-sample time series imputation, such as PriSTI [38] and [170], where the imputation has been regarded as a generation task. Refer to *Appendix D.1* for more details, available online.

### B. Out-of-Sample Imputation

To date, only a few GNN-based methods fall into this category. Among these works, IGNNK [73] proposes an inductive GNN kriging model to recover signals for unobserved time series, such as a new variable in a multivariate time series. In IGNNK, the training process involves masked subgraph sampling and signal reconstruction with the diffusion graph convolution network presented in [68]. Another similar work is SATCN [66], and the primary difference between these two works lies in the underlying GNN architectures. INCREASE [169], on the other hand, further considers heterogeneous spatial and diverse temporal relations among the locations, lifting the performance of inductive spatio-temporal kriging. Notably, GRIN [37] can handle both in-sample and out-of-sample imputations, as well as [165]. More discussion is in *Appendix D.2*, available online.

## VIII. PRACTICAL APPLICATIONS AND RESOURCES

Graph neural networks have been applied to a broad range of disciplines related to time series analysis. We categorize the mainstream applications of GNN4TS into seven areas: smart transportation, on-demand services, environment & sustainable energy, internet-of-things, physical systems, healthcare, and fraud detection. Three prominent areas in this section are discussed. A detailed expansion can be found in *Appendix E*, available online. In addition, we summarize the common benchmark datasets and the open-sourced implementation of representative models in *Appendix F*, available online.

*Smart Transportation:* The domain of transportation has been significantly transformed with the advent of GNNs, with typical applications spanning from traffic prediction to flight delay prediction. For example, by leveraging advanced algorithms and data analytics related to spatial-temporal GNNs, traffic conditions can be accurately predicted [175], [176], thereby facilitating efficient route planning and congestion management. Another important application is traffic data imputation, which is crucial for maintaining the integrity of traffic databases and ensuring the accuracy of traffic analysis and prediction models [85], [167]. There is also related research on autonomous driving [177] and flight delay prediction [178], [179]. The use of advanced technologies like GNNs in these applications underscores their transformative impact on smart transportation, emphasizing its critical role in evolving transportation systems.

*Environment & Sustainable Energy:* In this sector, GNNs have been instrumental in wind speed and power prediction, capturing the complex spatial-temporal dynamics of wind patterns to provide accurate predictions that aid in the efficient management of wind energy resources [180], [181]. Similarly, in solar energy, GNNs have been used for solar irradiance and photovoltaic (PV) power prediction, modeling the intricate relationships between various factors influencing solar energy generation to provide accurate predictions [182], [183]. Furthermore, GNNs have been employed for air pollution prediction [184] and weather forecasting [185] that are crucial for various services in agriculture, energy, and transportation.

*Physical Systems:* Systems of interacting objects are found in numerous scientific fields, including the simulation of n-body systems [186], particle physics [187], modeling of human motion dynamics [188], and prediction of molecular dynamics [189]. In graph-based deep learning, the objects are represented as nodes of a graph and GNNs have proven effective in modeling their complex interactions. Despite the challenges posed by physical constraints, promising performance has been achieved thanks to the incorporation of inductive biases from known physical laws [190], [191] and architectures that maintain

the symmetries of the underlying system, such as equivariance to rotations and translations [189], [192], [193].

*Other Applications:* The application of GNNs for time series analysis has also been extended to various other fields, such as finance [194], [195], on-demand services [41], fraud detection [196], manufacturing [197], and recommender systems [198]. As research in this area continues to evolve, it is anticipated that the application of GNN4TS will continue to expand, opening up new possibilities for data-driven decision making and system optimization.

## IX. Future Directions

The future of GNNs for time series analysis holds immense promise, driven by several key directions and challenges that need to be addressed to unlock their full potential. More detailed discussion is in *Appendix G*, available online.

*Pre-Training, Transfer Learning, and Large Models:* Pre-training, transfer learning, and large models are emerging as potent strategies to bolster the performance of GNNs in time series analysis [61], [199], especially when data is sparse or diverse. These techniques hinge on utilizing learned representations from one or more domains to enhance performance in other related domains [3]. Notable examples include Panagopoulos et al.'s meta-learning for COVID-19 prediction in data-limited settings [200], and Shao et al.'s pre-training framework for spatial-temporal GNNs [61]. The exploration of GNN pre-training and transferability for time series tasks is growing, especially with the advent of generative AI and large models capable of addressing diverse tasks [201]. Challenges include limited time series data for large-scale pre-training, ensuring wide coverage and transferability, and designing strategies to capture complex spatial-temporal dependencies [202].

*Robustness:* Robustness of GNNs refers to their ability to handle data perturbations and distribution shifts, especially those engineered by adversaries [203]. This is crucial for time series from dynamic systems, as operational failures can compromise the entire system's integrity [204]. For example, inadequate handling of noise or data corruption in smart city applications can disrupt traffic management, and in healthcare, it can cause missed critical treatment periods. While GNNs perform well in many applications, enhancing their robustness and developing effective failure management strategies are essential.

*Privacy Enhancing:* As GNNs become integral to various sectors, privacy protection is increasingly important. GNNs' capability to learn and reconstruct relationships within complex systems necessitates safeguarding the privacy of both individual entities (nodes) and their relationships (edges) in time series data. The interpretability of GNNs, while useful for identifying vulnerabilities, can also expose sensitive information [205]. Therefore, maintaining robust privacy defenses while capitalizing on the benefits of GNNs for time series analysis requires a delicate balance, one that calls for constant vigilance and continual innovation.

*Scalability:* While GNNs are effective in analyzing complex time series data, their adaptation to vast time-dependent data volumes presents challenges like memory constraints during computations. Traditional GNNs use sampling strategies such as node-wise [206], layer-wise [207], and graph-wise [208] to mitigate these issues, but preserving temporal dependencies is complex. Enhancing scalability for real-time GNN applications, especially on edge devices with limited computing power, is crucial. This intersection of scalability and efficient inference is a significant research area with potential for major advancements.

## X. Conclusion

This comprehensive survey bridges the knowledge gap in graph neural networks for time series analysis (GNN4TS) by reviewing recent advancements and offering a unified taxonomy to categorize existing works. It covers a wide range of analytical tasks, including forecasting, classification, anomaly detection, and imputation, providing a detailed understanding of current progress. We also delve into the intricacies of spatial and temporal dependencies modeling and overall model architecture, offering a fine-grained classification of individual studies. Highlighting the expanding applications of GNN4TS, we demonstrate its versatility and potential for future growth. This survey serves as a valuable resource for practitioners and experts, proposing future research directions to guide and inspire further work in GNN4TS.

## References

[1] Q. Wen, L. Yang, T. Zhou, and L. Sun, "Robust time series analysis and applications: An industrial perspective," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 4836–4837.
[2] P. Esling and C. Agon, "Time-series data mining," *ACM Comput. Surv.*, vol. 45, no. 1, pp. 1–34, 2012.
[3] K. Zhang et al., "Self-supervised learning for time series analysis: Taxonomy, progress, and prospects," 2023, *arXiv:2306.10125*.
[4] B. Lim and S. Zohren, "Time-series forecasting with deep learning: A survey," *Philos. Trans. Roy. Soc. A*, vol. 379, no. 2194, 2021, Art. no. 20200209.
[5] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: A review," *Data Mining Knowl. Discov.*, vol. 33, no. 4, pp. 917–963, 2019.
[6] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, "A review on outlier/anomaly detection in time series data," *ACM Comput. Surv.*, vol. 54, no. 3, pp. 1–33, 2021.
[7] C. Fang and C. Wang, "Time series data imputation: A survey on deep learning approaches," 2020, *arXiv: 2011.11347*.
[8] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1624–1639, Dec. 2011.
[9] Y. Zhou, Z. Ding, Q. Wen, and Y. Wang, "Robust load forecasting towards adversarial attacks via Bayesian learning," *IEEE Trans. Power Syst.*, vol. 38, no. 2, pp. 1445–1459, Mar. 2023.
[10] A. A. Cook, G. Mısırlı, and Z. Fan, "Anomaly detection for IoT time-series data: A survey," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6481–6494, Jul. 2020.
[11] S. Wang, J. Cao, and S. Y. Philip, "Deep learning for spatio-temporal data mining: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 8, pp. 3681–3700, Aug. 2022.
[12] J. Ye, J. Zhao, K. Ye, and C. Xu, "How to build a graph-based deep learning architecture in traffic domain: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 5, pp. 3904–3924, May 2022.
[13] W. Jiang and J. Luo, "Graph neural network for traffic forecasting: A survey," *Expert Syst. Appl.*, vol. 207, 2022, Art. no. 117921.
[14] K.-H. N. Bui, J. Cho, and H. Yi, "Spatial-temporal graph neural network for traffic forecasting: An overview and open research issues," *Appl. Intell.*, vol. 52, no. 3, pp. 2763–2774, 2022.
[15] G. Jin, Y. Liang, Y. Fang, J. Huang, J. Zhang, and Y. Zheng, "Spatio-temporal graph neural networks for predictive learning in urban computing: A survey," 2023, *arXiv:2303.14483*.

[16] Z. A. Sahili and M. Awad, "Spatio-temporal graph neural networks: A survey," 2023, *arXiv:2301.10569*.

[17] S. Rahmani, A. Baghbani, N. Bouguila, and Z. Patterson, "Graph neural networks for intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 8, pp. 8846–8885, Aug. 2023.

[18] A. Cini, I. Marisca, D. Zambon, and C. Alippi, "Taming local effects in graph-based spatiotemporal forecasting," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2023, Art. no. 2417.

[19] L.-J. Cao and F. E. H. Tay, "Support vector machine with adaptive parameters in financial time series forecasting," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 14, no. 6, pp. 1506–1518, Nov. 2003.

[20] Y. Xia and J. Chen, "Traffic flow forecasting method based on gradient boosting decision tree," in *Proc. 5th Int. Conf. Front. Manuf. Sci. Measuring Technol.*, Atlantis Press, 2017, pp. 413–416.

[21] B. Biller and B. L. Nelson, "Modeling and generating multivariate time-series input processes using a vector autoregressive technique," *ACM Trans. Model. Comput. Simul.*, vol. 13, no. 3, pp. 211–237, 2003.

[22] G. E. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *J. Amer. Statist. Assoc.*, vol. 65, no. 332, pp. 1509–1526, 1970.

[23] M. Jin, Y. Zheng, Y.-F. Li, S. Chen, B. Yang, and S. Pan, "Multivariate time series forecasting with dynamic graph neural ODEs," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 9, pp. 9168–9180, Sep. 2023.

[24] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu, "Convolutional neural networks for time series classification," *J. Syst. Eng. Electron.*, vol. 28, no. 1, pp. 162–169, 2017.

[25] A. Borovykh, S. Bohte, and C. W. Oosterlee, "Conditional time series forecasting with convolutional neural networks," 2017, *arXiv: 1703.04691*.

[26] J. T. Connor, R. D. Martin, and L. E. Atlas, "Recurrent neural networks and robust time series prediction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 5, no. 2, pp. 240–254, Mar. 1994.

[27] Q. Wen et al., "Transformers in time series: A survey," in *Proc. Int. Joint Conf. Artif. Intell.*, 2023, pp. 6778–6786.

[28] M. Jin et al., "How expressive are spectral-temporal graph neural networks for time series forecasting?," 2023, *arXiv:2305.06587*.

[29] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[30] M. Yang, M. Zhou, M. Kalander, Z. Huang, and I. King, "Discrete-time temporal network embedding via implicit hierarchical learning in hyperbolic space," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 1975–1985.

[31] M. Yang, M. Zhou, H. Xiong, and I. King, "Hyperbolic temporal network embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 11, pp. 11489–11502, Nov. 2023.

[32] H. Y. Koh, A. T. N. Nguyen, S. Pan, L. T. May, and G. I. Webb, "Physicochemical graph neural network for learning protein–ligand interaction fingerprints from sequence data," *Nature Mach. Intell.*, vol. 6, pp. 673–687, 2024.

[33] X. Zhang, M. Zeman, T. Tsiligkaridis, and M. Zitnik, "Graph-guided network for irregularly sampled multivariate time series," in *Proc. Int. Conf. Learn. Representations*, 2022.

[34] Z. Wang, T. Jiang, Z. Xu, J. Gao, and J. Zhang, "Irregularly sampled multivariate time series classification: A graph learning approach," *IEEE Intell. Syst.*, vol. 38, no. 3, pp. 3–11, May/Jun. 2023.

[35] H. Zhao et al., "Multivariate time-series anomaly detection via graph attention network," in *Proc. IEEE Int. Conf. Data Mining*, 2020, pp. 841–850.

[36] A. Deng and B. Hooi, "Graph neural network-based anomaly detection in multivariate time series," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 4027–4035.

[37] A. Cini, I. Marisca, and C. Alippi, "Filling the G_ap_s: Multivariate time series imputation by graph neural networks," in *Proc. Int. Conf. Learn. Representations*, 2022.

[38] M. Liu, H. Huang, H. Feng, L. Sun, B. Du, and Y. Fu, "PriSTI: A conditional diffusion framework for spatiotemporal imputation," in *Proc. IEEE Int. Conf. Data Eng.*, 2023, pp. 1927–1939.

[39] M. Jin, Y.-F. Li, and S. Pan, "Neural temporal walks: Motif-aware representation learning on continuous-time dynamic graphs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, Art. no. 1445.

[40] Y. Liu et al., "Graph self-supervised learning: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 6, pp. 5879–5900, Jun. 2023.

[41] X. Geng et al., "Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 3656–3663.

[42] S. He and K. G. Shin, "Towards fine-grained flow forecasting: A graph attention approach for bike sharing systems," in *Proc. Web Conf.*, 2020, pp. 88–98.

[43] X. Zhang, R. Cao, Z. Zhang, and Y. Xia, "Crowd flow forecasting with multi-graph neural networks," in *Proc. Int. Joint Conf. Neural Netw.*, 2020, pp. 1–7.

[44] M. Li and Z. Zhu, "Spatial-temporal fusion graph neural networks for traffic flow forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 4189–4196.

[45] S. Yi and V. Pavlovic, "Sparse Granger causality graphs for human action classification," in *Proc. Int. Conf. Pattern Recognit.*, 2012, pp. 3374–3377.

[46] Y. Wang, Z. Duan, Y. Huang, H. Xu, J. Feng, and A. Ren, "MTHetGNN: A heterogeneous graph embedding framework for multivariate time series forecasting," *Pattern Recognit. Lett.*, vol. 153, pp. 151–158, 2022.

[47] D. Grattarola, D. Zambon, L. Livi, and C. Alippi, "Change detection in graph streams by learning graph embeddings on constant-curvature manifolds," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 6, pp. 1856–1869, Jun. 2020.

[48] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph WaveNet for deep spatial-temporal graph modeling," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 1907–1913.

[49] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, Art. no. 1494.

[50] D. Cao et al., "Spectral temporal graph neural network for multivariate time-series forecasting," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, Art. no. 1491.

[51] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2020, pp. 753–763.

[52] C. Shang, J. Chen, and J. Bi, "Discrete graph structure learning for forecasting multiple time series," in *Proc. Int. Conf. Learn. Representations*, 2021.

[53] A. Cini, D. Zambon, and C. Alippi, "Sparse graph learning from spatiotemporal time series," *J. Mach. Learn. Res.*, vol. 24, no. 242, pp. 1–36, 2023.

[54] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 3634–3640.

[55] H. Wen et al., "DiffSTG: Probabilistic spatio-temporal graph forecasting with denoising diffusion models," in *Proc. 31st ACM Int. Conf. Adv. Geographic Inf. Syst.*, 2023, Art. no. 60.

[56] S. Han and S. S. Woo, "Learning sparse latent graph representations for anomaly detection in multivariate time series," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 2977–2986.

[57] E. Dai and J. Chen, "Graph-augmented normalizing flows for anomaly detection of multiple time series," in *Proc. Int. Conf. Learn. Representations*, 2022.

[58] D. Zambon, C. Alippi, and L. Livi, "Concept drift and anomaly detection in graph streams," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5592–5605, Nov. 2018.

[59] D. Zambon and C. Alippi, "Where and how to improve graph-based spatio-temporal predictors," 2023, *arXiv:2302.01701*.

[60] Y. Liu et al., "Multivariate time-series forecasting with temporal polynomial graph neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, Art. no. 1411.

[61] Z. Shao, Z. Zhang, F. Wang, and Y. Xu, "Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 1567–1577.

[62] Y. Zheng et al., "Correlation-aware spatial-temporal graph learning for multivariate time-series anomaly detection," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Nov. 14, 2023, doi: 10.1109/TNNLS.2023.3325667.

[63] D. Zha, K.-H. Lai, K. Zhou, and X. Hu, "Towards similarity-aware time-series classification," in *Proc. SIAM Int. Conf. Data Mining*, 2022, pp. 199–207.

[64] H. Liu et al., "TodyNet: Temporal dynamic graph neural network for multivariate time series classification," 2023, *arXiv:2304.05078*.

[65] Y. Ye, S. Zhang, and J. J. Yu, "Spatial-temporal traffic data imputation via graph attention convolutional network," in *Proc. Int. Conf. Artif. Neural Netw.*, 2021, pp. 241–252.

[66] Y. Wu, D. Zhuang, M. Lei, A. Labbe, and L. Sun, "Spatial aggregation and temporal convolution networks for real-time Kriging," 2021, *arXiv:2109.12144*.

[67] I. Marisca, A. Cini, and C. Alippi, "Learning to reconstruct missing data from spatiotemporal graphs with sparse observations," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, Art. no. 2324.

[68] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. Int. Conf. Learn. Representations*, 2018.

[69] W. Chen, L. Chen, Y. Xie, W. Cao, Y. Gao, and X. Feng, "Multi-range attentive bicomponent graph convolutional network for traffic forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 3529–3536.

[70] C. Zheng, X. Fan, C. Wang, and J. Qi, "GMAN: A graph multi-attention network for traffic prediction," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 1234–1241.

[71] Y. Chen, I. Segovia-Dominguez, and Y. R. Gel, "Z-GCNETs: Time zigzags at graph convolutional networks for time series forecasting," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 1684–1694.

[72] H. Yu et al., "Regularized graph structure learning with semantic knowledge for multi-variates time-series forecasting," in *Proc. Int. Joint Conf. Artif. Intell.*, 2022, pp. 2362–2368.

[73] Y. Wu, D. Zhuang, A. Labbe, and L. Sun, "Inductive graph neural networks for spatiotemporal Kriging," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 4478–4485.

[74] Z. Fang, Q. Long, G. Song, and K. Xie, "Spatial-temporal graph ODE networks for traffic flow forecasting," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 364–373.

[75] J. Choi, H. Choi, J. Hwang, and N. Park, "Graph neural controlled differential equations for traffic forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 6367–6374.

[76] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, "Urban traffic prediction from spatio-temporal data using deep meta learning," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2019, pp. 1720–1730.

[77] A. Cini, I. Marisca, F. Bianchi, and C. Alippi, "Scalable spatiotemporal graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 7218–7226.

[78] W. Hu, Y. Yang, Z. Cheng, C. Yang, and X. Ren, "Time-series event prediction with evolutionary state graph," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2021, pp. 580–588.

[79] W. Chen, L. Tian, B. Chen, L. Dai, Z. Duan, and M. Zhou, "Deep variational graph convolutional recurrent network for multivariate time series anomaly detection," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 3621–3633.

[80] L. Zhou, Q. Zeng, and B. Li, "Hybrid anomaly detection via multihead dynamic graph attention networks for multivariate time series," *IEEE Access*, vol. 10, pp. 40 967–40 978, 2022.

[81] Y. Liang, Z. Zhao, and L. Sun, "Memory-augmented dynamic graph convolution networks for traffic data imputation with diverse missing patterns," *Transp. Res. Part C Emerg.*, vol. 143, 2022, Art. no. 103826.

[82] K. Chen, M. Feng, and T. S. Wirjanto, "Multivariate time series anomaly detection via dynamic graph forecasting," 2023, *arXiv:2302.02051*.

[83] C. Yu, X. Ma, J. Ren, H. Zhao, and S. Yi, "Spatio-temporal graph transformer networks for pedestrian trajectory prediction," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2020, pp. 507–523.

[84] Y. Wu, M. Gu, L. Wang, Y. Lin, F. Wang, and H. Yang, "Event2Graph: Event-driven bipartite graph for multivariate time-series anomaly detection," 2021, *arXiv:2108.06783*.

[85] X. Wu, M. Xu, J. Fang, and X. Wu, "A multi-attention tensor completion network for spatiotemporal traffic data imputation," *IEEE Internet Things J.*, vol. 9, no. 20, pp. 20 203–20 213, Oct. 2022.

[86] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 922–929.

[87] X. Wang et al., "Traffic flow prediction via spatial temporal graph neural network," in *Proc. Web Conf.*, 2020, pp. 1082–1092.

[88] S. Lan, Y. Ma, W. Huang, W. Wang, H. Yang, and P. Li, "DSTAGNN: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 11 906–11 917.

[89] Z. Chen, D. Chen, X. Zhang, Z. Yuan, and X. Cheng, "Learning graph structures with transformer for multivariate time-series anomaly detection in IoT," *IEEE Internet Things J.*, vol. 9, no. 12, pp. 9179–9189, Jun. 2022.

[90] W. Zhang, C. Zhang, and F. Tsung, "GRELEN: Multivariate time series anomaly detection from the perspective of graph relational learning," in *Proc. Int. Joint Conf. Artif. Intell.*, 2022, pp. 2390–2397.

[91] Y. Xia et al., "Deciphering spatio-temporal graph forecasting: A causal lens and treatment," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2023, Art. no. 1611.

[92] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 914–921.

[93] Y. Cui et al., "METRO: A generic graph neural network framework for multivariate time series forecasting," in *Proc. VLDB Endowment*, vol. 15, no. 2, pp. 224–236, 2021.

[94] R.-G. Cirstea, B. Yang, and C. Guo, "Graph attention recurrent neural networks for correlated time series forecasting," in *Proc. 5th SIGKDD Workshop Mining Learn. Time Ser.*, 2019, pp. 1–6.

[95] Q. Zhang, J. Chang, G. Meng, S. Xiang, and C. Pan, "Spatio-temporal graph structure learning for traffic forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 1177–1185.

[96] Z. Cheng et al., "Time2Graph+: Bridging time series and graph representation learning via multiple attentions," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 2, pp. 2078–2090, Feb. 2023.

[97] W. Xi, A. Jain, L. Zhang, and J. Lin, "LB-SimTSC: An efficient similarity-aware graph neural network for semi-supervised time series classification," 2023, *arXiv:2301.04838*.

[98] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.

[99] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs," *IEEE Trans. Image Process.*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.

[100] C. Ying et al., "Do transformers really perform badly for graph representation?," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 28 877–28 888.

[101] C. Cai, T. S. Hy, R. Yu, and Y. Wang, "On the connection between MPNN and graph transformer," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 3408–3430.

[102] R. Wan, S. Mei, J. Wang, M. Liu, and F. Yang, "Multivariate temporal convolutional network: A deep neural networks approach for multivariate time series forecasting," *Electronics*, vol. 8, no. 8, 2019, Art. no. 876.

[103] J. Gao and B. Ribeiro, "On the equivalence between temporal and static equivariant graph representations," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 7052–7076.

[104] D. Zambon, D. Grattarola, C. Alippi, and L. Livi, "Autoregressive models for sequences of graphs," in *Proc. Int. Joint Conf. Neural Netw.*, 2019, pp. 1–8.

[105] R. Huang, C. Huang, Y. Liu, G. Dai, and W. Kong, "LSGCN: Long short-term traffic prediction with graph convolutional networks," in *Proc. Int. Joint Conf. Artif. Intell.*, 2020, pp. 2355–2361.

[106] B. Paassen, D. Grattarola, D. Zambon, C. Alippi, and B. E. Hammer, "Graph edit networks," in *Proc. Int. Conf. Learn. Representations*, 2021.

[107] J. Chauhan, A. Raghuveer, R. Saket, J. Nandy, and B. Ravindran, "Multi-variate time series forecasting on variable subsets," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 76–86.

[108] X. Rao, H. Wang, L. Zhang, J. Li, S. Shang, and P. Han, "FOGS: First-order gradient supervision with learning-based graph for traffic flow forecasting," in *Proc. Int. Joint Conf. Artif. Intell.*, 2022, pp. 3926–3932.

[109] A. Cini, D. Mandic, and C. Alippi, "Graph-based time series clustering for end-to-end hierarchical forecasting," 2023, *arXiv:2305.19183*.

[110] Z. Li, L. Xia, Y. Xu, and C. Huang, "GPT-ST: Generative pre-training of spatio-temporal graph neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2024, Art. no. 3077.

[111] G. Lai, W. Chang, Y. Yang, and H. Liu, "Modeling long- and short-term temporal patterns with deep neural networks," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2018, pp. 95–104.

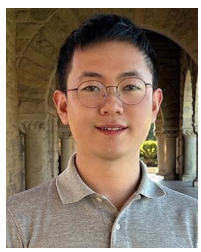
[112] S.-Y. Shih, F.-K. Sun, and H.-Y. Lee, "Temporal pattern attention for multivariate time series forecasting," *Mach. Learn.*, vol. 108, pp. 1421–1441, 2019.

[113] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 3837–3845.

[114] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1003–1012.

[115] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1263–1272.
[116] J. Klicpera, S. Weißenberger, and S. Günnemann, "Diffusion improves graph learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 13 333–13 345.
[117] Y. Zheng, H. Zhang, V. Lee, Y. Zheng, X. Wang, and S. Pan, "Finding the missing-half: Graph complementary learning for homophily-prone and heterophily-prone graphs," in *Proc. Int. Conf. Mach. Learn.*, 2023, Art. no. 1788.
[118] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *Proc. NeurIPS Workshop Deep Learn.*, 2014.
[119] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017.
[120] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018.
[121] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 3558–3565.
[122] A. Feng and L. Tassiulas, "Adaptive graph spatial-temporal transformer network for traffic forecasting," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2022, pp. 3933–3937.
[123] C. Wang, K. Zhang, H. Wang, and B. Chen, "Auto-STGCN: Autonomous spatial-temporal graph convolutional network search," *ACM Trans. Knowl. Discov. Data*, vol. 17, no. 5, pp. 1–21, 2023.
[124] A. Vaswani et al., "Attention is all you need," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[125] C. Park et al., "ST-GRAT: A novel spatio-temporal graph attention networks for accurately forecasting dynamically changing road speed," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 1215–1224.
[126] Z. Diao, X. Wang, D. Zhang, Y. Liu, K. Xie, and S. He, "Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 890–897.
[127] K. Guo, Y. Hu, Y. Sun, S. Qian, J. Gao, and B. Yin, "Hierarchical graph convolution network for traffic forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 151–159.
[128] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 6572–6583.
[129] P. Kidger, J. Morrill, J. Foster, and T. J. Lyons, "Neural controlled differential equations for irregular time series," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, Art. no. 562.
[130] S. Guan, B. Zhao, Z. Dong, M. Gao, and Z. He, "GTAD: Graph and temporal neural network for multivariate time series anomaly detection," *Entropy*, vol. 24, no. 6, 2022, Art. no. 759.
[131] S. S. Srinivas, R. K. Sarkar, and V. Runkana, "Hypergraph learning based recommender system for anomaly detection, control and optimization," in *Proc. IEEE Int. Conf. Big Data*, 2022, pp. 1922–1929.
[132] D. Zambon, L. Livi, and C. Alippi, "Graph iForest: Isolation of anomalous and outlier graphs," in *Proc. Int. Joint Conf. Neural Netw.*, 2022, pp. 1–8.
[133] W. Chen, Z. Zhou, Q. Wen, and L. Sun, "Time series subsequence anomaly detection via graph neural networks," 2023. [Online]. Available: https://openreview.net/forum?id=73U_NlKaNx
[134] D. M. Hawkins, *Identification of Outliers*, vol. 11. Berlin, Germany: Springer, 1980.
[135] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Process.*, vol. 99, pp. 215–249, 2014.
[136] Z. Z. Darban, G. I. Webb, S. Pan, C. C. Aggarwal, and M. Salehi, "Deep learning for time series anomaly detection: A survey," 2022, *arXiv:2211.05244*.
[137] E. Keogh, J. Lin, and A. Fu, "HOT SAX: Efficiently finding the most unusual time series subsequence," in *Proc. 5th IEEE Int. Conf. Data Mining*, 2005, Art. no. 8.
[138] K. M. Ting, B.-C. Xu, T. Washio, and Z.-H. Zhou, "Isolation distributional kernel a new tool for point & group anomaly detection," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 3, pp. 2697–2710, Mar. 2023.
[139] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an LSTM-based variational autoencoder," *IEEE Trans. Robot. Autom.*, vol. 3, no. 3, pp. 1544–1551, Jul. 2018.
[140] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Söderström, "Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2018, pp. 387–395.
[141] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2019, pp. 2828–2837.
[142] J. Xu, H. Wu, J. Wang, and M. Long, "Anomaly transformer: Time series anomaly detection with association discrepancy," in *Proc. Int. Conf. Learn. Representations*, 2022.
[143] M. Jin, Y. Liu, Y. Zheng, L. Chi, Y.-F. Li, and S. Pan, "ANEMONE: Graph anomaly detection with multi-scale contrastive learning," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2021, pp. 3122–3126.
[144] Y. Zheng, M. Jin, Y. Liu, L. Chi, K. T. Phan, and Y.-P. P. Chen, "Generative and contrastive self-supervised learning for graph anomaly detection," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12220–12233, Dec. 2023.
[145] A. Garg, W. Zhang, J. Samaran, R. Savitha, and C.-S. Foo, "An evaluation of anomaly detection and diagnosis in multivariate time series," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 6, pp. 2508–2517, Jun. 2022.
[146] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[147] M. Ranzato, Y. Boureau, and Y. LeCun, "Sparse feature learning for deep belief networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 1185–1192.
[148] D. P. Kingma et al., "An introduction to variational autoencoders," *Found. Trends Mach. Learn.*, vol. 12, no. 4, pp. 307–392, 2019.
[149] D. Zambon, C. Alippi, and L. Livi, "Change-point methods on a sequence of graphs," *IEEE Trans. Signal Process.*, vol. 67, no. 24, pp. 6327–6341, Dec. 2019.
[150] Y. Zheng et al., "Graph spatiotemporal process for multivariate time series anomaly detection with missing values," *Inf. Fusion*, vol. 106, 2024, Art. no. 102255.
[151] Z. Duan et al., "Multivariate time-series classification with hierarchical variational graph pooling," *Neural Netw.*, vol. 154, pp. 481–490, 2022.
[152] Z. Diao et al., "EC-GCN: A encrypted traffic classification framework based on multi-scale graph convolution networks," *Comput. Netw.*, vol. 224, 2023, Art. no. 109614.
[153] R. Younis, A. Hakmeh, and Z. Ahmadi, "MTS2Graph: Interpretable multivariate time series classification with temporal evolving graphs," *Pattern Recognit.*, vol. 152, 2024, Art. no. 110486.
[154] N. M. Foumani, L. Miller, C. W. Tan, G. I. Webb, G. Forestier, and M. Salehi, "Deep learning for time series classification and extrinsic regression: A current survey," 2023, *arXiv:2302.02515*.
[155] J. Lines and A. Bagnall, "Time series classification with ensembles of elastic distance measures," *Data Mining Knowl. Discov.*, vol. 29, pp. 565–592, 2015.
[156] M. Herrmann and G. I. Webb, "Amercing: An intuitive, elegant and effective constraint for dynamic time warping," 2021, *arXiv:2111.13314*.
[157] J. Lines, S. Taylor, and A. Bagnall, "Time series classification with HIVE-COTE: The hierarchical vote collective of transformation-based ensembles," *ACM Trans. Knowl. Discov. Data*, vol. 12, no. 5, 2018, Art. no. 52.
[158] M. Middlehurst, J. Large, M. Flynn, J. Lines, A. Bostrom, and A. Bagnall, "HIVE-COTE 2.0: A new meta ensemble for time series classification," *Mach. Learn.*, vol. 110, no. 11/12, pp. 3211–3243, 2021.
[159] A. Dempster, F. Petitjean, and G. I. Webb, "ROCKET: Exceptionally fast and accurate time series classification using random convolutional kernels," *Data Mining Knowl. Discov.*, vol. 34, no. 5, pp. 1454–1495, 2020.
[160] C. W. Tan, A. Dempster, C. Bergmeir, and G. I. Webb, "MultiRocket: Multiple pooling operators and transformations for fast and effective time series classification," *Data Mining Knowl. Discov.*, vol. 36, no. 5, pp. 1623–1646, 2022.
[161] Z. Cheng, Y. Yang, W. Wang, W. Hu, Y. Zhuang, and G. Song, "Time2Graph: Revisiting time series modeling with dynamic shapelets," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 3617–3624.
[162] P. Boniol and T. Palpanas, "Series2Graph: Graph-based subsequence anomaly detection for time series," in *Proc. VLDB Endowment*, vol. 13, no. 12, pp. 1821–1834, 2022.
[163] E. Keogh and C. A. Ratanamahatana, "Exact indexing of dynamic time warping," *Knowl. Inf. Syst.*, vol. 7, pp. 358–386, 2005.
[164] S. Tang et al., "Self-supervised graph neural networks for improved electroencephalographic seizure analysis," in *Proc. Int. Conf. Learn. Representations*, 2022.
[165] A. Roth and T. Liebig, "Forecasting unobserved node states with spatio-temporal graph neural networks," 2022, *arXiv:2211.11596*.

[166] Y. Chen, Z. Li, C. Yang, X. Wang, G. Long, and G. Xu, "Adaptive graph recurrent network for multivariate time series imputation," in *Proc. Int. Conf. Neural Inf. Process.*, 2023, pp. 64–73.
[167] X. Kong, W. Zhou, G. Shen, W. Zhang, N. Liu, and Y. Yang, "Dynamic graph convolutional recurrent imputation network for spatiotemporal traffic missing data," *Knowl.-Based Syst.*, vol. 261, 2023, Art. no. 110188.
[168] G. Shen, W. Zhou, W. Zhang, N. Liu, Z. Liu, and X. Kong, "Bidirectional spatial-temporal traffic data imputation via graph attention recurrent neural network," *Neurocomputing*, vol. 531, pp. 151–162, 2023.
[169] C. Zheng, X. Fan, C. Wang, J. Qi, C. Chen, and L. Chen, "INCREASE: Inductive graph representation learning for spatio-temporal Kriging," in *Proc. ACM Web Conf.*, 2023, pp. 673–683.
[170] T. Yun, H. Jung, and J. Son, "Imputation as inpainting: Diffusion models for spatiotemporal data imputation," 2023. [Online]. Available: https://openreview.net/forum?id=QUANtQnx30l
[171] S. Moritz and T. Bartz-Beielstein, "imputeTS: Time series missing value imputation in R," *R J.*, vol. 9, no. 1, 2017, Art. no. 207.
[172] M. Saad, M. Chaudhary, F. Karray, and V. Gaudet, "Machine learning based approaches for imputation in time series data and their impact on forecasting," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, 2020, pp. 2621–2627.
[173] 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, 2018, Art. no. 6085.
[174] X. Miao, Y. Wu, J. Wang, Y. Gao, X. Mao, and J. Yin, "Generative semi-supervised learning for multivariate time series imputation," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 8983–8991.
[175] Y. Tang, A. Qu, A. H. Chow, W. H. Lam, S. Wong, and W. Ma, "Domain adversarial spatial-temporal network: A transferable framework for short-term traffic forecasting across cities," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2022, pp. 1905–1915.
[176] H. Li et al., "Traffic flow forecasting in the COVID-19: A deep spatial-temporal model based on discrete wavelet transformation," *ACM Trans. Knowl. Discov. Data*, vol. 17, no. 5, pp. 1–28, 2023.
[177] L. Tang, F. Yan, B. Zou, W. Li, C. Lv, and K. Wang, "Trajectory prediction for autonomous driving based on multiscale spatial-temporal graph," *IET Intell. Transp. Syst.*, vol. 17, no. 2, pp. 386–399, 2023.
[178] K. Cai, Y. Li, Y.-P. Fang, and Y. Zhu, "A deep learning approach for flight delay prediction through time-evolving graphs," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 11 397–11 407, Aug. 2022.
[179] Z. Guo et al., "SGDAN—A spatio-temporal graph dual-attention neural network for quantified flight delay prediction," *Sensors*, vol. 20, no. 22, 2020, Art. no. 6433.
[180] Q. Wu, H. Zheng, X. Guo, and G. Liu, "Promoting wind energy for sustainable development by precise wind speed prediction based on graph neural networks," *Renewable Energy*, vol. 199, pp. 977–992, 2022.
[181] Y. He, S. Chai, J. Zhao, Y. Sun, and X. Zhang, "A robust spatio-temporal prediction approach for wind power generation based on spectral temporal graph neural network," *IET Renewable Power Gener.*, vol. 16, no. 12, pp. 2556–2565, 2022.
[182] X. Jiao, X. Li, D. Lin, and W. Xiao, "A graph neural network based deep learning predictor for spatio-temporal group solar irradiance forecasting," *IEEE Trans. Ind. Inform.*, vol. 18, no. 9, pp. 6142–6149, Sep. 2022.
[183] M. Zhang et al., "Optimal graph structure based short-term solar PV power forecasting method considering surrounding spatio-temporal correlations," *IEEE Trans. Ind Appl.*, vol. 59, no. 1, pp. 345–357, Jan./Feb. 2023.
[184] V. O. Santos, P. A. C. Rocha, J. Scott, J. Van Griensven Thé, and B. Gharabaghi, "Spatiotemporal air pollution forecasting in Houston-TX: A case study for ozone using deep graph neural networks," *Atmosphere*, vol. 14, no. 2, 2023, Art. no. 308.
[185] R. Keisler, "Forecasting global weather with graph neural networks," 2022, *arXiv:2202.07575*.
[186] P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende, and K. Kavukcuoglu, "Interaction networks for learning about objects, relations and physics," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 4502–4510.
[187] G. Shi, D. Zhang, M. Jin, and S. Pan, "Towards complex dynamic physics system simulation with graph neural ODEs," 2023, *arXiv:2305.12334*.
[188] Y. Liu, S. Magliacane, M. Kofinas, and E. Gavves, "Graph switching dynamical systems," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 21867–21883.
[189] L. Wu, Z. Hou, J. Yuan, Y. Rong, and W. Huang, "Equivariant spatio-temporal attentive graph networks to simulate physical dynamics," in *Proc. 37th Int. Conf. Neural Inf. Process. Syst.*, 2023, Art. no. 1965.
[190] A. Sanchez-Gonzalez, V. Bapst, K. Cranmer, and P. Battaglia, "Hamiltonian graph networks with ODE integrators," 2019, *arXiv: 1909.12790*.
[191] Y. Liu et al., "SEGNO: Generalizing equivariant graph neural networks with physical inductive biases," in *Proc. Int. Conf. Learn. Representations*, 2023.
[192] V. G. Satorras, E. Hoogeboom, and M. Welling, "E(n) equivariant graph neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 9323–9332.
[193] J. Brandstetter, R. Hesselink, E. van der Pol, E. J. Bekkers, and M. Welling, "Geometric and physical quantities improve E(3) equivariant message passing," in *Proc. Int. Conf. Learn. Representations*, 2021.
[194] J. Wang, S. Zhang, Y. Xiao, and R. Song, "A review on graph neural network methods in financial applications," *J. Data Sci.*, vol. 20, no. 2, pp. 111–134, 2022.
[195] Z. Song, Y. Zhang, and I. King, "Towards fair financial services for all: A temporal GNN approach for individual fairness on transaction networks," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2023, pp. 2331–2341.
[196] N. Noorshams, S. Verma, and A. Hofleitner, "TIES: Temporal interaction embeddings for enhancing social media integrity at Facebook," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2020, pp. 3128–3135.
[197] X.-Y. Yao, G. Chen, M. Pecht, and B. Chen, "A novel graph-based framework for state of health prediction of lithium-ion battery," *J. Energy Storage*, vol. 58, 2023, Art. no. 106437.
[198] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 346–353.
[199] X. Wang, D. Wang, L. Chen, and Y. Lin, "Building transportation foundation model via generative graph transformer," 2023, *arXiv:2305.14826*.
[200] G. Panagopoulos, G. Nikolentzos, and M. Vazirgiannis, "Transfer graph neural networks for pandemic forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 4838–4845.
[201] M. Jin et al., "Large models for time series and spatio-temporal data: A survey and outlook," 2023, *arXiv:2310.10196*.
[202] W. X. Zhao et al., "A survey of large language models," 2023, *arXiv:2303.18223*.
[203] H. Zhang, B. Wu, X. Yuan, S. Pan, H. Tong, and J. Pei, "Trustworthy graph neural networks: Aspects, methods and trends," in *Proc. IEEE*, vol. 112, no. 2, pp. 97–139, Feb. 2024.
[204] G. Dong et al., "Graph neural networks in IoT: A survey," *ACM Trans. Sensor Netw.*, vol. 19, no. 2, pp. 1–50, 2023.
[205] J. Xu, M. Xue, and S. Picek, "Explainability-based backdoor attacks against graph neural networks," in *Proc. 3rd ACM Workshop Wireless Secur. Mach. Learn.*, 2021, pp. 31–36.
[206] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.
[207] J. Chen, T. Ma, and C. Xiao, "FastGCN: Fast learning with graph convolutional networks via importance sampling," in *Proc. Int. Conf. Learn. Representations*, 2018.
[208] W. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C. Hsieh, "Cluster-GCN: An efficient algorithm for training deep and large graph convolutional networks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 257–266.



**Ming Jin** received the PhD degree from Monash University, Australia, in 2024. He is currently an assistant professor with the School of Information and Communication Technology, Griffith University. He has published more than twenty peer-reviewed papers in top-ranked journals and conferences, including *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *IEEE Transactions on Knowledge and Data Engineering*, NeurIPS, ICLR, ICML, etc. He serves as an associate editor of the *Journal of Neurocomputing* and regularly contributes as area chair/PC member for major AI conferences. His research interests include time series analysis, graph neural networks, and multi-modal learning.

**Huan Yee Koh** received the master of data science and bachelors of commerce degrees from Monash University, Melbourne, Australia, in 2021 and 2018, respectively. He is currently working toward the PhD degree in machine learning with Monash University. His research focuses on graph neural networks, time series analysis, drug discovery, data mining, and machine learning.

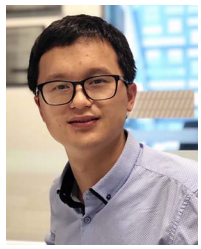
**Qingsong Wen** received the PhD degree in electrical and computer engineering from the Georgia Institute of Technology. He is the head of AI Research & Chief Scientist, Squirrel Ai Learning. He has published more than 100 top-ranked AI conference and journal papers, had multiple Oral/Spotlight Papers at NeurIPS, ICML, and ICLR, had multiple Most Influential Papers at IJCAI, received multiple IAAI Deployed Application Awards at AAAI, and won First Place of SP Grand Challenge at ICASSP. He organizes workshops on AI for Time Series and AI for education and serves as an associate editor of the *Neurocomputing* and *IEEE Signal Processing Letters*, and guest editor of the *Applied Energy* and *IEEE Internet of Things Journal*. His research focuses on AI for time series, AI for education, and general machine learning.

**Daniele Zambon** received the PhD degree from the Università della Svizzera italiana. He is a postdoctoral researcher with the Swiss AI Lab IDSIA, Università della Svizzera Italiana (Switzerland). He has been a visiting researcher/intern with the University of Florida (US), the University of Exeter (U.K.), and STMicroelectronics (Italy). He is a member of the IEEE CIS Task Force on Learning for Graphs and has co-organized special sessions and tutorials on deep learning and graph data. His main research interests include graph representation learning, time series analysis, and learning in non-stationary environments.

**Cesare Alippi** (Fellow, IEEE) is a professor with the Università della Svizzera italiana and Politecnico di Milano, a visiting professor with the University of Guangzhou, and an advisory professor with Northwestern Polytechnic, Xi'an. He has authored/coauthored one monograph, seven edited books, and around 200 papers, and holds eight patents. His research focuses on adaptation and learning in non-stationary environments, graph learning, intelligence for embedded systems, IoT, and cyber-physical systems. He is a fellow of the European Laboratory for Learning and Intelligent Systems, serves on the IEEE Computational Intelligence Society (CIS) Administrative Committee, and is a board of governors member of the International Neural Network Society. He has held various roles within IEEE CIS, including vice-president for education and awards committee chair, and has received several awards, including the International Neural Networks Society Gabor Award.

**Geoffrey I. Webb** (Fellow, IEEE) is a professor with the Monash University Department of Data Science and AI, Australia. He was editor in chief of the *Leading Data Mining Journal*, *Data Mining and Knowledge Discovery*, from 2005 to 2014. He has been program committee chair of both the leading data mining conferences, ACM SIGKDD and IEEE ICDM, as well as general chair of ICDM. He is a technical advisor to the startups BigML Inc and FROOMLE. He developed many of the key mechanisms of support-confidence association discovery in the 1980s. He pioneered multiple research areas as diverse as black-box user modelling, interactive data analytics, and statistically-sound pattern discovery. His many awards include IEEE ICDM Ten-Year Impact Award (2023) and the inaugural Eureka Prize for Excellence in Data Science.

**Irwin King** (Fellow, IEEE) received the BS degree in computer science from Caltech, and the PhD degree in computer science from the University of Southern California. He is professor with the Department of Computer Science & Engineering, Chinese University of Hong Kong. His research interests include machine learning, social computing, AI, web intelligence, data mining, and multimedia information processing, with more than 300 technical publications in these areas. He is an associate editor of the *Journal of Neural Networks*, an ACM distinguished member, and a fellow of the International Neural Network Society (INNS), and Hong Kong Institute of Engineers (HKIE). He has served as president of the International Neural Network Society (INNS) and as general co-chair of several major conferences. Currently, he is the vice-chair of ACM SIGWEB and WebConf Steering Committee. He has received the ACM CIKM 2019 Test of Time Award, the ACM SIGIR 2020 Test of Time Award, and the 2020 APNNS Outstanding Achievement Award for contributions to social computing with machine learning.



**Shirui Pan** (Senior Member, IEEE) received the PhD degree in computer science from the University of Technology Sydney (UTS), Ultimo, NSW, Australia. He is a professor with the School of Information and Communication Technology, Griffith University, Australia. Prior to this, he was a senior lecturer with the Faculty of IT, Monash University. His research interests include data mining and machine learning. To date, he has published more than 100 research papers in top-tier journals and conferences, including *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Neural Networks and Learning Systems*, ICML, NeurIPS, and KDD. His research received the 2024 CIS IEEE TNNLS Oustanding Paper Award and the 2020 IEEE ICDM Best Student Paper Award. He is recognized as one of the AI 2000 AAAI/IJCAI Most Influential Scholars in Australia. He is an ARC future fellow and a fellow of Queensland Academy of Arts and Sciences (FQA).