Contents lists available at ScienceDirect

Applied and Computational Harmonic Analysis

journal homepage: www.elsevier.com/locate/acha

# Graph signal processing on dynamic graphs based on temporal-attention product [☆]

Ru Geng [a], Yixian Gao [a], Hong-Kun Zhang [b], Jian Zu [a,*]

[a] *Center for Mathematics and Interdisciplinary Sciences, and School of Mathematics and Statistics, Northeast Normal University, Changchun, 130024, China*
[b] *Department of Mathematics and Statistics, University of Massachusetts, Amherst, MA 01003, USA*

ARTICLE INFO

*Article history:*
Received 25 August 2022
Received in revised form 19 July 2023
Accepted 23 July 2023
Available online 28 July 2023
Communicated by Naoki Saito

*Keywords:*
Time-varying data
Signal processing
Spectral graph wavelet transform
Visual analysis
Graph Fourier transform
Graph attention neural network
Dynamic networks

ABSTRACT

Signal processing is an important research topic. This paper aims to provide a general framework for signal processing on arbitrary dynamic graphs. We propose a new graph transformation by defining a temporal-attention product. This product transforms the sequence of graph time slices with arbitrary topology and number of nodes into a static graph, effectively capturing graph signals' spatio-temporal dynamic evolution process. The temporal-attention product graph provides a solid mathematical foundation to model the time-dependent graph signal processes as martingales. The weighted adjacency matrix obtained by temporal-attention products is a block tridiagonal matrix, which has been extensively studied. Therefore, it is general and convenient to perform graph signal processing on this new static graph. We apply two real datasets to illustrate the effectiveness of spectral graph wavelet transform based on temporal-attention product. For one of the datasets with no graph structure, we learn the graph weights through a neural network.

## 1. Introduction

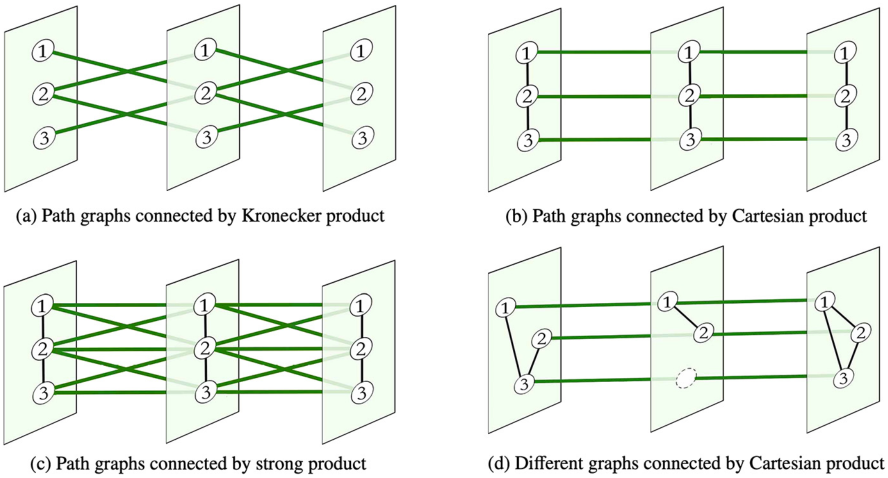
Graph signal processing (GSP) is an emerging field in data science, and it has received much attention in many fields, such as classifying cancer types, temporal brain data, theoretical chemistry, social network analysis, computer networks (such as the Internet) and distributed systems, etc. Analyzing graph signal data will help us understand the behavior patterns in the network, which is crucial in several application

[☆] YX Gao is partially supported by NSFC grants (11871140, 12071065) and National Key R&D Program of China (2020YFA0714102). HK Zhang is partially supported by Simons Foundation Collaboration Grants for Mathematicians (706383). J Zu is partially supported by NSFC grants (11971096, 11971095).

[*] Corresponding author.

*E-mail addresses:* gengru93@163.com (R. Geng), gaoyx643@nenu.edu.cn (Y. Gao), hongkun@math.umass.edu (H.-K. Zhang), zuj100@nenu.edu.cn (J. Zu).

https://doi.org/10.1016/j.acha.2023.101579
1063-5203/© 2023 Elsevier Inc. All rights reserved.



(a) Path graphs connected by Kronecker product

(b) Path graphs connected by Cartesian product

(c) Path graphs connected by strong product

(d) Different graphs connected by Cartesian product

**Fig. 1.** Graph products. The green line is the temporal edge. The dotted line circle is a manually added node. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

areas, including sensor network data, processing and analysis of biological data, and applications of image processing and machine learning [1], etc.

Many powerful tools have been proposed for studying graph signals, e.g., the graph Fourier transform (GFT) [2–5], and the windowed graph Fourier transform (WGFT) [6,7]. However, these methods can not capture local relationships of graph signals well and fail to identify abrupt signal changes. Due to its multiresolution advantages, the spectral graph wavelet transforms (SGWT) proposed by Hammond et al. [8] and Shuman et al. [6,9,10] is an improved tool to perform visualization analysis, detect and characterize the attributes of signals, and play an important role in node classifications. For instance, Mohan et al. [11] take the vehicle speed as signals and apply SGWT to detect the occurrence, propagation, and span of destructive events such as traffic congestion, to guide and plan traffic routes. Other applications include community mining [12], visual analysis [13], surface denoising [14], research on manifolds [15], etc.

The following entities are involved in the definitions of graphs: $V$ (a set of nodes), $E$ (a set of edges), $f$ (signals defined on nodes), and $w$ (weights defined on edges). A dynamic graph is obtained when any of these four entities change over time [16]. Graphs in many real-world applications are inherently dynamic, such as data-packet traffic on the Internet, disease spreading on social networks, temperature changes in an area, users in e-commerce platforms continuing to interact with new items and connections established in a communication network over time, etc. Because of the time-varying property of dynamic graphs, existing GSP methods are severely hampered, and tools such as GFT, WGFT, and SGWT cannot be directly applied to dynamic graphs.

One could use the graph product structure to obtain a static graph. The three well-known graph products are the Kronecker product, the Cartesian product, and the strong product. These methods are useful in studying discrete temporal graphs, where the graph time slices $G_t$ have identical nodes and edges, see [8,13,17–20]. Fig. 1 (a), (b) and (c) depict the three graph products using the path graph[1] of three nodes as an example. Another method is to perform GFT by expressing the Laplace of the dynamic graph as a tensor, and obtaining the transformed basis function by Tucker decomposition of the tensor [21]. Alternatively, the Laplace operator of the dynamic graph is represented as a discrete second-order derivative in time, and then GFT and SGWT are performed [22]. All these methods are designed to deal with graph signals on sequences of graph slices with the same nodes or topology. Recently, an important attempt was made in [23], where the authors manually added some additional isolated points on graph time slices, to ensure all

[1] A path graph is a graph whose nodes are adjacent to exactly two other nodes, with the exception of the two extreme ones that are connected to only one node.

graph time slices have identical nodes, see Fig. 1 (d) (The dotted line circle is a manually added node). They then connected these graphs by Cartesian product and used SGWT for the visual analysis of signals. However, in real-world applications, this process can be rather expensive and unrealistic, as it not only adds time edges that carry no information (such as on those manually added, isolated nodes) but also needs to change the previous topology connection every time a new graph time slice is added. Another issue is that by adding the extra temporal edges of the same nodes in adjacent graph time slices, the underlying diffusion mechanism changes as if there are self-loops in each graph time slice.

To overcome these difficulties and effectively capture the temporal evolution of signals in dynamic graphs, inspired by the attention mechanism introduced by Bahdanau et al. [24] in machine learning for language processing, we propose to add time edges that capture the best information similarity carried by nodes in adjacent graph time slices. We call this way of adding temporal edges the temporal-attention product. As a result, a discrete dynamic graph with $T$ graph time slices (snapshots) $\{G_t, t = 1, \cdots, T\}$ is transformed into a static graph $\mathcal{G}_T$, which is called a transformed graph. The nested transformed graphs $\{\mathcal{G}_t, t = 1, \cdots, T\}$ accurately capture both the spatial and temporal structure of the first $T$ discrete dynamic graph snapshots $\{G_t, t = 1, \cdots T\}$, and are also suited for studying the dynamic evolution process of the graph signals. We can define a filtration of $\sigma$-algebra $\{\mathcal{F}_t, t \geq 1\}$ generated by graph signals on $\{\mathcal{G}_t, t \geq 1\}$. The transformed graph $\mathcal{G}_t$ also provides a general mathematical tool for modeling graph signal processes using advanced methods in probability theory (including diffusion and martingale processes, etc.). Furthermore, this new construction is inductive: to construct $\mathcal{G}_{t+1}$ at each new time step, $t+1$, one only adds temporal edges for nodes between $G_t$ and $G_{t+1}$ based on the graph structure of $G_t$. The detailed construction of the transformed graphs can be found in section 2.

The weighted adjacency matrix $\mathcal{W}_T$ of the transformed graph is a generic symmetrical block tridiagonal matrix. The block tridiagonal matrix can be found in many applications in the finite difference method [25,26], discrete Sturm-Liouville operators [27], discrete transport problem simulation and electronic structure calculations [28–30], random walks and birth-and-death processes [31,32], scattering theory [33], computational fluid dynamics [34], signal processing [25,35,36] and so on. It is convenient to study the spectrum of the transformed graphs since the properties of block tridiagonal matrices have been extensively studied. A widely used direct method is to compute eigenvalues and eigenvectors based on divide-and-conquer [37–42], or twisted block factorizations [43,44]. For some special cases, the relationship between tridiagonal matrices and orthogonal polynomials can be used to obtain eigenvalues and eigenvectors [45–49]. In this paper, we will give some spectral properties of the weighted adjacency matrix of the transformed graph and some recursive formulas for GSP on the transformed graph.

SGWT is powerful in our transformed graph $\mathcal{G}_t$, enabling spatio-temporal anomaly detection and multiresolution visual signal analysis. Two real-world datasets are used to show that SGWT coefficients on the transformed graph accurately capture the spatio-temporal dynamic changes of the signals. In one of our applications, we only have multivariate time series. The underlying graph for these time series is unknown prior and can not be constructed preliminarily through the topology of nodes of the graph. In this paper, we explore a deep learning method to learn the graph link weights. More precisely, we apply the graph attention neural network (GAT), a powerful graph neural network that introduces the attention mechanism to refine the convolution process in a generic graph convolutional neural network [50].

This paper is organized as follows: In section 2, we introduce the temporal-attention product on dynamic graphs. In section 3, we give the spectral properties of undirected dynamic graphs. In section 4, we discuss GFT and SGWT on dynamic graphs. In section 5, we introduce the classification method based on SGWT coefficients. In section 6, we analyze two real-world datasets using spectral graph wavelet visualization. We close with a conclusion section.

**Table 1**
Notations in $G_t$ and $\mathcal{G}_t$.

| $G_t$ | $\mathcal{G}_t$ |
|---|---|
| $V_t$: Node set on graph $G_t$; | $\mathcal{V}_t$: Node set on graph $\mathcal{G}_t$; |
| $N_t$: Cardinality of $V_t$, i.e., $\mid V_t \mid= N_t$; | $N_t^{\#}$: Cardinality of $\mathcal{V}_t$, i.e., $\mid \mathcal{V}_t \mid= N_t^{\#}$; |
| $E_t$: Edge set on graph $G_t$; | $\mathcal{E}_t$: Edge set on graph $\mathcal{G}_t$; |
| | $E_{t,t+1} \in \mathcal{E}_t$: Temporal-attention edge set between $G_t$ and $G_{t+1}$; |
| $(v,t) \in V_t$: Node on the $t$-th time slice $G_t$; | $\mathbf{v}$: Node on $\mathcal{G}_t$; |
| $e_{(v,w,t)} \in E_t$: Edge joining $(v,t)$ and $(w,t)$; | $\mathbf{e}_{(\mathbf{v},\mathbf{w})} \in \mathcal{E}_t$: Edge joining $\mathbf{v}$ and $\mathbf{w}$; |
| $\mathbf{A}_t = (a_{(v,w,t)}) \in \mathbb{R}^{N_t \times N_t}$: Adjacency matrix of $G_t$. | $\mathcal{A}_t = (\mathbf{a}_{(\mathbf{v},\mathbf{w})}) \in \mathbb{R}^{N_t^{\#} \times N_t^{\#}}$: Adjacency matrix of $\mathcal{G}_t$. |
| $a_{(v,w,t)} = 1$ if there exists $e_{(v,w,t)} \in E_t$; otherwise, $a_{(v,w,t)} = 0$; | $\mathbf{a}_{(\mathbf{v},\mathbf{w})} = 1$ if there exists $\mathbf{e}_{(\mathbf{v},\mathbf{w})} \in \mathcal{E}_t$; otherwise, $\mathbf{a}_{(\mathbf{v},\mathbf{w})} = 0$; |
| $\mathbf{W}_t = (w_{(v,w,t)}) \in \mathbb{R}^{N_t \times N_t}$: Weighted adjacency matrix. | $\mathcal{W}_t = (\mathbf{w}_{(\mathbf{v},\mathbf{w})}) \in \mathbb{R}^{N_t^{\#} \times N_t^{\#}}$: Weighted adjacency matrix. |
| $w_{(v,v,t)} \in \mathbb{R}_+$ represents the link intensity between $(v,t)$ and $(w,t)$; | $\mathbf{w}_{(\mathbf{v},\mathbf{w})} \in \mathbb{R}_+$ represents the link intensity between $\mathbf{v}$ and $\mathbf{w}$; |
| $N^1_{(v,t)} = \{(w,t) \in V_t \mid e_{(v,w,t)} \in E_t\}$: One-hop neighbor of $(v,t)$ on $G_t$. | $\mathcal{N}_{\mathbf{v}}^{1,t} = \{\mathbf{w} \in \mathcal{V}_t \mid \mathbf{e}_{(\mathbf{v},\mathbf{w})} \in \mathcal{E}_t\}$: Spatio-temporal one-hop neighbor of $\mathbf{v}$ on $\mathcal{G}_t$. |

### 1.1. Notation

Table 1 lists the notations used in the graph time slice $G_t$ and the transformed graph $\mathcal{G}_t$.

## 2. The temporal-attention product on dynamic graphs

Signal processing on static graphs is an important research topic and has been applied in many tasks over the years. However, most networks are dynamic in real applications, and their structures or properties are constantly changing over time. Possible changes include the insertion and deletion of nodes (objects), insertion and deletion of edges (relationships), and modification of attributes (for example, the node's signal or the weight of the edge). A discrete dynamic graph consists of $T$ graph snapshots (time slices), which are observed along with the evolution of a dynamic graph. Specifically, the $T$ graph snapshots can be denoted as $\{G_t, t = 1, \cdots, T\}$, where $G_t$ is the graph observed at time $t$. In these cases, one major question is to analyze the spatio-temporal behavior of graph signals $\{f_t, t = 1, \cdots, T\}$ defined on the graph slice sequence $\{G_t, t = 1, \cdots, T\}$. To capture the temporal evolution in the sequence of graphs, we introduce a new graph topology, by defining a temporal-attention product for studying dynamic graph networks.

### 2.1. Definition of temporal-attention product on dynamic graphs

We consider a discrete dynamic graph network, represented by a sequence of undirected graph time slices $\{G_t = (V_t, E_t), t = 1, \cdots, T\}$. The same node may appear in different time slices. We call the set of all different nodes in the $T$ time slices the base node set $B_T$. We denote $(v,t) \in V_t$, $t \leq T$ if there is a base node $v \in B_T$ on the graph time slice $G_t$. $E_t$ is an edge set with the element $e_{(v,w,t)}$ connecting $(v,t)$ to $(w,t)$. Let $(v,t) \in V_t$ and $(v,t+1) \in V_{t+1}$, which represent the same base node $v$ on the adjacent time slice. We present a new method by introducing the temporal-attention product of graphs, which enables us to transform these graph time slices into a static graph $\mathcal{G}_t$.

**Definition 1.** Let $(v,t) \in V_t$, $(w,t+1) \in V_{t+1}$. We say $\mathbf{e}_{((v,t),(w,t+1))}$ is a temporal-attention edge connecting $(v,t)$ to $(w,t+1)$, if there exists $(w,t) \in V_t$ belonging to the one-hop neighbor of $(v,t)$ in $V_t$. The temporal-attention product $G_t \times_{at} G_{t+1}$ is defined by adding all temporal-attention edges between $G_t$ and $G_{t+1}$.

See Fig. 2 for an example, where green lines are temporal-attention edges. Graph time slices are the same as Fig. 1 (d).

Now we can introduce how to construct a static graph $\mathcal{G}_t$ inductively. Let $N_t = |V_t|$ be the cardinality of $V_t$. We first define $\mathcal{G}_1 = G_1$ and denote $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{E}_1)$. The node elements in $\mathcal{V}_1$ inherit from $V_1$ are

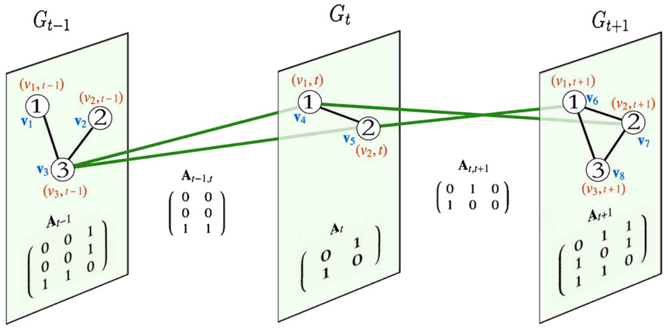


**Fig. 2.** A dynamic model of the transformed graph using temporal-attention product. Green lines are temporal-attention edges. $\mathbf{A}_{t-1}$, $\mathbf{A}_t$ and $\mathbf{A}_{t+1}$ are the adjacency matrices of $G_{t-1}$, $G_t$ and $G_{t+1}$ respectively.

denoted again by $(v, 1)$. Let the bold-type letter $\mathbf{e}_{((v,1),(w,1))}$ be the edge of $\mathcal{E}_1$ between $(v, 1)$ and $(w, 1)$, which inherits from $e_{(v,w,1)} \in E_1$. For the next time slice, let $G_2 = (V_2, E_2)$. We can define $\mathcal{G}_2 = \mathcal{G}_1 \times_{at} G_2$ with the node set $\mathcal{V}_2 = \mathcal{V}_1 \cup V_2$, $|\mathcal{V}_2| = N_1 + N_2$ and $\mathcal{E}_2 = \mathcal{E}_1 \cup E_2 \cup E_{1,2}$, where $E_{1,2}$ is the collection of temporal-attention edges from $G_1 \times_{at} G_2$. Assume inductively $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$ has been well defined.

**Definition 2.** We say $\mathcal{G}_{t+1} = (\mathcal{V}_{t+1}, \mathcal{E}_{t+1})$ is a transformed graph if $\mathcal{G}_{t+1} = \mathcal{G}_t \times_{at} G_{t+1}$ satisfying $\mathcal{V}_{t+1} = \mathcal{V}_t \cup V_{t+1}$, and $\mathcal{E}_{t+1} = \mathcal{E}_t \cup E_{t+1} \cup E_{t,t+1}$, where $E_{t,t+1}$ is the collection of temporal-attention edges from $G_t \times_{at} G_{t+1}$.

In graph theory, an adjacency matrix is a classical matrix representation for a graph, that allows us to establish certain graph properties using matrix-theoretic methods. Its rows and columns correspond to graph nodes and are both indexed by identical node orderings. Let $\mathbf{A}_t = (a(v, w, t))$ be the adjacency matrix of $G_t$ with $a(v, w, t) = 1$ if there exists $e(v, w, t) \in E_t$; otherwise, $a(v, w, t) = 0$. To concisely represent nodes in the transformed graph $\mathcal{G}_T$, we can use bold-type letters to relabel the nodes. Let $\mathcal{A}_t = (\mathbf{a}_{(\mathbf{v}_i, \mathbf{v}_j)})$ be the adjacency matrix of $\mathcal{G}_t$ with $\mathbf{a}(\mathbf{v}_i, \mathbf{v}_j) = 1$ if there exists $\mathbf{e}(\mathbf{v}_i, \mathbf{v}_j) \in \mathcal{E}_t$; otherwise, $a(\mathbf{v}_i, \mathbf{v}_j) = 0$.

Let $N_t^{\#}$ be the cardinality of $\mathcal{V}_t$ in the transformed graph $\mathcal{G}_t$. The order of the rows and columns of the adjacency matrix $\mathcal{A}_t$ is arranged in increasing order of graph time slices. This order relation allows for the flexible addition of new time slices to the transformed graph.

Obviously, the adjacency matrix $\mathcal{A}_{t+1} \in \mathbb{R}^{N_{t+1}^{\#} \times N_{t+1}^{\#}}$ satisfies:

$$\mathcal{A}_{t+1} = \begin{pmatrix} \mathcal{A}_t & \mathcal{A}_{t,t+1} \\ \mathcal{A}'_{t,t+1} & \mathbf{A}_{t+1} \end{pmatrix}, \tag{1}$$

where $\mathcal{A}_{t,t+1} \in \mathbb{R}^{N_t^{\#} \times N_{t+1}}$ infers whether there are edges between $\mathcal{G}_t$ and $G_{t+1}$, and $\mathcal{A}'_{t,t+1}$ denotes the transpose of $\mathcal{A}_{t,t+1}$.

**Theorem 1.** *Let $\mathcal{G}_{t+1} = (\mathcal{V}_{t+1}, \mathcal{E}_{t+1})$ be a transformed graph with an adjacency matrix $\mathcal{A}_{t+1}$. Then $\mathcal{A}_{t+1}$ is a block tridiagonal matrix and $\mathcal{A}_{t,t+1}$ has the following expression:*

$$\mathcal{A}_{t,t+1} = \begin{pmatrix} \mathbf{0} \\ \mathbf{A}_{t,t+1} \end{pmatrix}, \tag{2}$$

*where $\mathbf{0} \in \mathbb{R}^{N_{t-1}^{\#} \times N_{t+1}}$ denotes a matrix that contains all 0 elements and $\mathbf{A}_{t,t+1} \in \mathbb{R}^{N_t \times N_{t+1}}$ describes whether there are edges between $G_t$ and $G_{t+1}$. Here $\mathbf{A}_{t,t+1}$ is obtained based on $\mathbf{A}_t$ by:*

*(i). deleting the i-th column if $(v, t)$ is the i-th node of $G_t$ and $(v, t+1)$ does not belong to $G_{t+1}$;*

*(ii). inserting a zero-filled column as the $j$-th column if $(v,t)$ is the $j$-th node of $G_{t+1}$ and $(v,t)$ does not belong to $G_t$.*

**Proof.** By Definition 1, the temporal-attention edges are defined between two adjacent time slices, we have $\mathbf{a}_{(\mathbf{v}_i,\mathbf{v}_j)} = 0$ for any $\mathbf{v}_i \in \mathcal{V}_{t-1}$ and $\mathbf{v}_j \in V_{t+1}$. Thus, the first $N^{\#}_{t-1}$ rows of $\mathcal{A}_{t,t+1}$ must be zero which implies that $\mathcal{A}_{t+1}$ is a block tridiagonal matrix. The last $N_t$ rows of $\mathcal{A}_{t,t+1}$ can be nonzero, which is just right $\mathbf{A}_{t,t+1}$. Its row index corresponds to the nodes of $G_t$, and its column index corresponds to the nodes of $G_{t+1}$.

Define

$$\Gamma[(v_i,t)] = \begin{cases} (v_i,t) & \text{if the base node } v_i \text{ belongs to } G_t, \\ \text{NULL} & \text{otherwise.} \end{cases}$$

Let the node orderings of $G_t$ be $\Gamma[(v_1,t)], \cdots, \Gamma[(v_{N_b},t)]$, $t \geq 1$, where $N_b$ is the cardinality of the base set $B_T$. Let $(v_i,t) \in V_t$ and $(v_i,t+1) \in V_{t+1}$, which represent the same base node $v_i$ on the adjacent time slice. Next, we can prove that $\mathbf{A}_{t,t+1}$ is based on $\mathbf{A}_t$.

Let $(v_l,t) \in V_t$ and $(v_r,t) \in V_t$. If $(v_l,t+1) \in V_{t+1}$, we have $\mathbf{a}_{((v_r,t),(v_l,t+1))} = a_{(v_r,v_l,t)}$. Obviously, the column of $\mathbf{A}_{t,t+1}$ corresponding to $(v_l,t+1)$ is equal to the column of $\mathbf{A}_t$ corresponding to $(v_l,t)$. Let $(v,t)$ be the $i$-th node of $G_t$. If $(v,t+1)$ does not belong to $G_{t+1}$, $\mathbf{A}_{t,t+1}$ doesn't inherit the $i$-th column of $\mathbf{A}_t$. Let $(w,t+1)$ be the $j$-th node of $G_{t+1}$. If $(w,t)$ does not belong to $G_t$, there is no edge between $(w,t+1)$ and any node of $G_t$. Because of the existence of the column of $\mathbf{A}_{t+1}$ corresponding to node $(w,t+1)$, we should insert a zero-filled column as the $j$-th column of $\mathbf{A}_{t,t+1}$. □

We next present an example to illustrate Theorem 1 using Fig. 2. Fig. 2 depicts three graph time slices $G_{t-1}$, $G_t$, and $G_{t+1}$ that are connected by the temporal-attention product. As a comparison, their Cartesian product connection has been shown in Fig. 1 (d). From the structure of graphs $G_{t-1}$, $G_t$, and $G_{t+1}$, we have

$$\mathbf{A}_{t-1} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}, \quad \mathbf{A}_t = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \mathbf{A}_{t+1} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}.$$

Compared to $G_{t-1}$, $G_t$ keeps the base nodes $v_1$ and $v_2$, so $\mathbf{A}_{t-1,t}$ is obtained by keeping the 1-st and 2-nd columns of $\mathbf{A}_{t-1}$, i.e.,

$$\mathbf{A}_{t-1,t} = \begin{pmatrix} 0 & 0 & \cancel{1} \\ 0 & 0 & \cancel{1} \\ 1 & 1 & \cancel{0} \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{pmatrix}.$$

By Theorem 1, we get

$$\mathcal{A}_t = \begin{pmatrix} \mathcal{A}_{t-2} & \mathcal{A}_{t-2,t-1} & \\ \mathcal{A}'_{t-2,t-1} & \mathbf{A}_{t-1} & \mathbf{A}_{t-1,t} \\ & \mathbf{A}'_{t-1,t} & \mathbf{A}_t \end{pmatrix} = \left(\begin{array}{c|ccc|cc} \mathcal{A}_{t-2} & \multicolumn{3}{c|}{\mathcal{A}_{t-2,t-1}} & & \\ \hline & 0 & 0 & 1 & 0 & 0 \\ \mathcal{A}'_{t-2,t-1} & 0 & 0 & 1 & 0 & 0 \\ & 1 & 1 & 0 & 1 & 1 \\ \hline & 0 & 0 & 1 & 0 & 1 \\ & 0 & 0 & 1 & 1 & 0 \end{array}\right),$$

where the blank space represents the zero matrices. Similarly, $G_{t+1}$ has one more base node $v_3$ than $G_t$, so $\mathbf{A}_{t,t+1}$ is obtained by adding a zero-filled column in $\mathbf{A}_t$'s 3-rd column, i.e.,

$$\mathbf{A}_{t,t+1} = (\mathbf{A}_t \ \mathbf{0}) = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$

Therefore,

$$\mathcal{A}_{t+1} = \begin{pmatrix} \mathcal{A}_{t-2} & \mathcal{A}_{t-2,t-1} & & \\ \mathcal{A}'_{t-2,t-1} & \mathbf{A}_{t-1} & \mathbf{A}_{t-1,t} & \\ & \mathbf{A}'_{t-1,t} & \mathbf{A}_t & \mathbf{A}_{t,t+1} \\ & & \mathbf{A}'_{t,t+1} & \mathbf{A}_{t+1} \end{pmatrix}.$$

### *2.2. Martingale approximations of graph signals*

Now we consider a graph signal function $f : \cup_{t=1}^{T} V_t \to \mathbb{R}$ defined on the sequence of graph time slices $\{G_t, t = 1, \cdots, T\}$. Using the temporal-attention products, $\{G_t, t = 1, \cdots, T\}$ has now been transformed into a sequence of increasing graphs $\{\mathcal{G}_t, t = 1, \cdots, T\}$. Using the fact that $\mathcal{V}_T = \cup_{t=1}^{T} V_t$, thus the graph signal $f$ has a natural extension on the transformed graph $\mathcal{G}_T$, with $f : \mathcal{V}_T \to \mathbb{R}$. In real-world applications, even if $T < \infty$ represents the current time, we would also like to consider future time slices, with $T + 1, T + 2, \cdots$. We denote $\mathcal{V}_\infty = \lim_{t\to\infty} \mathcal{V}_t$. One important task for GSP is to perform forecasting, and one would like to guarantee the stability of the graph signals processing method with increasing graph time slices. This is impossible for classical GSP on the original dynamic graph network $\{G_t, t = 1, \cdots, T\}$, as $G_{T+1}$ is completely unknown. One advantage of our transformed graphs is that the analysis of graph signals on the transformed graphs becomes more stable as time slices increase. Let $L^2(\mathcal{V}_T)$ be the collection of all real-valued square-summable functions defined on $\mathcal{V}_T$. The following result shows that the transformed graph guarantees a Martingale approximation $\{f_t, t \geq 0\}$ of a possibly partially observed graph signal $f$ by the time $T$.

**Theorem 2.** *For any graph signal $f \in L^2(\mathcal{V}_\infty)$, defined on the original dynamic graph network $\{G_t, t \geq 1\}$, there is a martingale approximation sequence $\{f_t, t \geq 0\}$, such that*

$$\lim_{t\to\infty} f_t = f \tag{3}$$

*converges almost surely.*

**Proof.** Let $\{\mathcal{G}_t, t \geq 1\}$ be the sequence of transformed graphs for a dynamic graph network $\{G_t, t \geq 1\}$. Let $\mathcal{F} = \sigma(L^2(\mathcal{V}_\infty))$ be the $\sigma$-algebra on $\mathcal{V}_\infty$ generated by the space of square-summable functions. Let $\mu$ be any probability measure defined on $(\mathcal{V}_\infty, \mathcal{F})$, which is absolutely continuous with respect to the Lebesgue measure.

Next, we define a sequence of $\sigma$-algebra $\mathcal{F}_t = \sigma(L^2(\mathcal{V}_t))$, $t \geq 1$, which is generated by all $L^2$ functions (graph signals) on the transformed graph $\mathcal{G}_t$. And $\mathcal{F}_0$ is the trivial $\sigma$-algebra. Since $\mathcal{V}_t \subset \mathcal{V}_{t+1}$ is an increasing sequence of subsets in $\mathcal{V}_\infty$, one can check easily that the collection $\{\mathcal{F}_t, t \geq 0\}$ is a filtration on the probability space $(\mathcal{V}_\infty, \mathcal{F}, \mu)$.

Given any graph signal $f \in L^2(\mathcal{V}_\infty)$ defined on the original discrete dynamic graph network $\{G_t, t \geq 1\}$, we define, for $t = 1, 2 \cdots$,

$$f_t = \mathbb{E}(f|\mathcal{F}_t) \tag{4}$$

to be the conditional expectation of $f$ on the $\sigma$-algebra $\mathcal{F}_t$. Moreover, $f_0 = \mathbb{E}(f)$ to be the expectation of $f$ on $\mathcal{V}_\infty$. We claim that $\{f_t, t \geq 0\}$ is a Martingale with respect to the filtration $\{\mathcal{F}_t, t \geq 0\}$. Note that for any $t \geq 0$, and $s \geq 1$,

$$\mathbb{E}(f_{t+s}|\mathcal{F}_t) = \mathbb{E}(\mathbb{E}(f|\mathcal{F}_{t+s})|\mathcal{F}_t) = \mathbb{E}(f|\mathcal{F}_t) = f_t,$$

where we use the tower property for conditional expectations in the last step. Thus we have shown that $\{f_t, t = 0, 1, 2, \cdots\}$ is a martingale. Note that $f_t \in L^2(\mathcal{V}_t)$ and $\sup_t \mathbb{E}(|f_t|^2) < \infty$. Using Doob's Martingale Convergence Theorem, we know that $f_t \to f$ almost surely, as $t \to \infty$. □

Going back to the forecasting task, even if we can obtain the best prediction $f_T$ on each time slice $T$, one may not know what a graph signal $f : \mathcal{V}_\infty \to \mathbb{R}$ looks like, nor does not know it even exists. But our Theorem 3 guarantees that $\lim_{T\to\infty} f_T$ exists, almost surely.

**Theorem 3.** *Let $\{\mathcal{G}_t, t \geq 1\}$ be an increasing sequence of transformed graphs. Let $f_t \in L^2(\mathcal{V}_t)$ be the best-forecasted graph signal at time $t$, i.e., $\{f_t, t \geq 0\}$ is an $L^2$ bounded Martingale with respect to the filtration $\{\mathcal{F}_t = \sigma(L^2(\mathcal{V}_t)), t \geq 0\}$ on $\mathcal{V}_\infty$. Then there exists a graph signal $f \in L^2(\mathcal{V}_\infty)$, such that*

$$\lim_{t\to\infty} f_t = f \tag{5}$$

*converges almost surely and in $L^2(\mathcal{V}_\infty)$.*

**Proof.** Since $f \in L^2(\mathcal{V}_t)$, we can directly apply Doob's Martingale Convergence. □

Now we can see that the transformed graph has not only practical significance, but also lays a solid mathematical foundation for us to model the graph signal processes, using advanced tools in Martingale theorems.

## 3. Spectral properties for weighted dynamic graph network

We consider a discrete undirected dynamic graph network, which is represented by a sequence of graph time slices $\{G_t = (V_t, E_t), t = 1, \cdots, T\}$ with a weighted adjacency matrix $\mathbf{W}_t = (w_{(v,w,t)})$. The element $w_{(v,w,t)} \in \mathbb{R}_+$ is the weight relationship between $(v, t)$ and $(w, t)$. In particular, if $a_{(v,w,t)} = 0$, we have $w_{(v,w,t)} = 0$.

By the temporal-attention product, we have a transformed graph $\mathcal{G}_T = (\mathcal{E}_T, \mathcal{V}_T)$ with a weighted adjacency matrix $\mathcal{W}_T = (\mathbf{w}_{(\mathbf{v}_i,\mathbf{v}_j)})$. The element $\mathbf{w}_{(\mathbf{v}_i,\mathbf{v}_j)} \in \mathbb{R}_+$ encodes how strong the relationship between $\mathbf{v}_i$ and $\mathbf{v}_j$. Let $\mathbf{w}_{(\mathbf{v}_i,\mathbf{v}_j)} = 0$ if $\mathbf{a}_{(\mathbf{v}_i,\mathbf{v}_j)} = 0$ in the adjacency matrix. The weighted adjacency matrix of $\mathcal{G}_T$ is

$$\mathcal{W}_T = \begin{pmatrix} \mathbf{W}_1 & \mathbf{W}_{1,2} & & & \\ \mathbf{W}'_{1,2} & \mathbf{W}_2 & \mathbf{W}_{2,3} & & \\ & \mathbf{W}'_{2,3} & \mathbf{W}_3 & \ddots & \\ & & \ddots & \ddots & \mathbf{W}_{T-1,T} \\ & & & \mathbf{W}'_{T-1,T} & \mathbf{W}_T \end{pmatrix}, \tag{6}$$

where $\mathcal{W}_T \in \mathbb{R}^{(N_T^\# \times N_T^\#)}$, $\mathbf{W}_t \in \mathbb{R}^{N_t \times N_t}$, $\mathbf{W}_{t-1,t} \in \mathbb{R}^{N_{t-1} \times N_t}$ describes the weights of edges between $G_{t-1}$ and $G_t$. $\mathbf{W}'_{t-1,t}$ denotes the transpose of a matrix $\mathbf{W}_{t-1,t}$. The blank position in the matrix represents that filled with 0. Since $\mathcal{W}_T$ is symmetric, its eigenvalues and eigenvectors can be calculated using the divide-and-conquer method [39] or the twisted block factorizations method [43]. Next, we discuss eigenvalues and eigenvectors of the transformed graph in some special cases.

### *3.1. Case 1: $G_t$ shares the same $V_1$, $E_1$, and $\mathbf{W}_{t,t+1} = \mathbf{W}_t = \mathbf{W}_1$ for all $t \geq 1$*

We consider when the topological structure of the graphs on each time slice is identical. More precisely, there exists a fixed node set $V_1$ with $| V_1 |= N$ (edge set $E_1$), such that any $V_t$ ($E_t$) shares the same set of base nodes in $V_1$ (base edges in $E_1$). This is equivalent to assuming that the transformed graph $G_T = (V_T, E_T)$ defined by the temporal-attention product satisfies the following:

**Assumption (h).** *The transformed graph $\mathcal{G}_T$ has the weighted adjacency matrix $\mathcal{W}_T \in \mathbb{R}^{NT\times NT}$ satisfying $\mathbf{W}_{t,t+1} = \mathbf{W}_t = \mathbf{W}_1 \in \mathbb{R}^{N\times N}$.*

$\mathbf{W}_1$ has a completed set of eigenvalues $\{\lambda_n, n = 1, \cdots, N\}$ and orthogonal eigenvectors $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$. Let $\mathbf{H} = (\mathbf{H}_{i,j})$ be the adjacency matrix of the path graph with $T$ nodes, i.e.,

$$\mathbf{H}_{i,j} = \begin{cases} 1 & \text{if } |\mathrm{i}-\mathrm{j}| = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

The eigenvalues and eigenvectors corresponding to $\mathbf{H}$ are $\{\mu_1, \mu_2 \cdots \mu_T\}$ and $\{\mathbf{y}_1, \mathbf{y}_2 \cdots \mathbf{y}_T\}$, respectively. Specifically, they are $\mu_i = 2\cos(i\pi/(T+1))$, and $\mathbf{y}_i(j) = \sin((ij\pi)/(T+1))$, see [51], where $\mathbf{y}_i(j)$ denotes the $j$-th element of the eigenvector corresponding to the $i$-th eigenvalue.

Using the spectral information from $\mathbf{H}$ and $\mathbf{W}_1$, we can investigate the spectral properties of the transformed graph $\mathcal{G}_T$. First, we introduce the definition of Kronecker product, see [52].

**Definition 3.** Given two matrices $\mathbf{C} = (c_{ij})_{m_1\times m_2}$ and $\mathbf{B} = (b_{ij})_{n_1\times n_2}$, the Kronecker product of $\mathbf{C}$ and $\mathbf{B}$ is defined by

$$\mathbf{C}\otimes\mathbf{B} = \begin{pmatrix} c_{11}\mathbf{B} & c_{12}\mathbf{B} & \cdots & c_{1m_2}\mathbf{B} \\ c_{21}\mathbf{B} & c_{22}\mathbf{B} & \cdots & c_{2m_2}\mathbf{B} \\ \vdots & \vdots & \cdots & \vdots \\ c_{m_11}\mathbf{B} & c_{m_12}\mathbf{B} & \cdots & c_{m_1m_2}\mathbf{B} \end{pmatrix}_{(m_1n_1)\times(m_2n_2)}.$$

Some necessary properties of the Kronecker product are stated as the following lemma, see [52]:

**Lemma 4.** *Let $\mathbf{B}_1 \in \mathbb{R}^{m\times n}$, $\mathbf{B}_2 \in \mathbb{R}^{s\times r}$, $\mathbf{B}_3 \in \mathbb{R}^{n\times p}$, $\mathbf{B}_4 \in \mathbb{R}^{r\times t}$, and $k \in \mathbb{R}$. Then*
*(i).* $k(\mathbf{B}_1\otimes\mathbf{B}_2) = k\mathbf{B}_1\otimes\mathbf{B}_2 = \mathbf{B}_1\otimes k\mathbf{B}_2$,
*(ii).* $(\mathbf{B}_1+\mathbf{B}_2)\otimes\mathbf{B}_3 = \mathbf{B}_1\otimes\mathbf{B}_3 + \mathbf{B}_2\otimes\mathbf{B}_3$,
*(iii).* $(\mathbf{B}_1\otimes\mathbf{B}_2)\otimes\mathbf{B}_3 = \mathbf{B}_1\otimes(\mathbf{B}_2\otimes\mathbf{B}_3)$,
*(iv).* $(\mathbf{B}_1\otimes\mathbf{B}_2)(\mathbf{B}_3\otimes\mathbf{B}_4) = (\mathbf{B}_1\mathbf{B}_3)\otimes(\mathbf{B}_2\mathbf{B}_4)$.

The absence of operators in $(iv)$ corresponds to the usual matrix product. Next, we obtain a theorem describing the eigenvalues and eigenvectors of the transformed graph.

**Theorem 5.** *Let $\mathcal{G}_T$ be the transformed graph satisfying* **Assumption (h)***. Its weighted adjacency matrix can be expressed as $\mathcal{W}_T = \mathbf{W}_1\otimes\mathbf{H} + \mathbf{W}_1\otimes\mathbf{I}_T$, where $\otimes$ is Kronecker product. $\mathbf{H}$ is the adjacency matrix of the path graph with $T$ nodes, $\mathbf{I}_T$ is the identity matrix of size $T\times T$. Moreover, the eigenvalues and eigenvectors of the transformed graph are $\lambda_r\mu_s + \lambda_r$ and $\mathbf{x}_r\otimes\mathbf{y}_s$ $(r = 1, 2\cdots N; s = 1, 2\cdots T)$, respectively.*

**Proof.** Based on (6) and Assumption (h), the weighted adjacency matrix of the transformed graph $\mathcal{G}_T$ is

$$\mathcal{W}_T = \begin{pmatrix} \mathbf{W}_1 & \mathbf{W}_1 & & & \\ \mathbf{W}_1 & \mathbf{W}_1 & \mathbf{W}_1 & & \\ & \mathbf{W}_1 & \mathbf{W}_1 & \ddots & \\ & & \ddots & \ddots & \mathbf{W}_1 \\ & & & \mathbf{W}_1 & \mathbf{W}_1 \end{pmatrix}_{(NT\times NT)},$$

where

$$\mathbf{W}_1 = \begin{pmatrix} w_{(v_1,v_1,1)} & w_{(v_1,v_2,1)} & \cdots & w_{(v_1,v_N,1)} \\ w_{(v_2,v_1,1)} & w_{(v_2,v_2,1)} & \cdots & w_{(v_2,v_N,1)} \\ \vdots & \vdots & \cdots & \vdots \\ w_{(v_N,v_1,1)} & w_{(v_N,v_2,1)} & \cdots & w_{(v_N,v_N,1)} \end{pmatrix}_{(N\times N)}$$

with $w_{(v_i,v_j,1)} = w_{(v_j,v_i,1)}$. We reorder nodes according to the sequence of node time series. More precisely, we change the order of rows and columns in matrix $\mathcal{W}_T$ from

$$(v_1,1),\ (v_2,1),\ \cdots,\ (v_N,1),\ (v_1,2),\ (v_2,2),\ \cdots,\ (v_N,2),\ (v_1,T),\ (v_2,T),\ \cdots,\ (v_N,T)$$

to

$$(v_1,1),\ (v_1,2),\ \cdots,\ (v_1,T),\ (v_2,1),\ (v_2,2),\ \cdots,\ (v_2,T),\ (v_N,1),\ (v_N,2),\ \cdots,\ (v_N,T).$$

The resulting matrix is denoted again as $\mathcal{W}_T$ with

$$\begin{pmatrix} \tilde{\mathbf{W}}_{11} & \tilde{\mathbf{W}}_{12} & \tilde{\mathbf{W}}_{13} & \cdots & \tilde{\mathbf{W}}_{1N} \\ \tilde{\mathbf{W}}_{21} & \tilde{\mathbf{W}}_{22} & \tilde{\mathbf{W}}_{23} & \cdots & \tilde{\mathbf{W}}_{2N} \\ \tilde{\mathbf{W}}_{31} & \tilde{\mathbf{W}}_{32} & \tilde{\mathbf{W}}_{33} & \cdots & \tilde{\mathbf{W}}_{3N} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \tilde{\mathbf{W}}_{N1} & \tilde{\mathbf{W}}_{N2} & \tilde{\mathbf{W}}_{N3} & \cdots & \tilde{\mathbf{W}}_{NN} \end{pmatrix}_{(NT\times NT)},$$

where

$$\tilde{\mathbf{W}}_{ij} = \begin{pmatrix} w_{(v_i,v_j,1)} & w_{(v_i,v_j,1)} & & & \\ w_{(v_i,v_j,1)} & w_{(v_i,v_j,1)} & w_{(v_i,v_j,1)} & & \\ & w_{(v_i,v_j,1)} & w_{(v_i,v_j,1)} & \ddots & \\ & & \ddots & \ddots & w_{(v_i,v_j,1)} \\ & & & w_{(v_i,v_j,1)} & w_{(v_i,v_j,1)}, \end{pmatrix}_{(T\times T)}, \quad i,\ j \in \{1,2,\cdots N\}.$$

By Definition 3 and (7), we have $\mathcal{W}_T = \mathbf{W}_1 \otimes \mathbf{H} + \mathbf{W}_1 \otimes \mathbf{I}_T$. Since $\mathbf{W}_1\mathbf{x}_r = \lambda_r \mathbf{x}_r$, $\mathbf{H}\mathbf{y}_s = \mu_r \mathbf{y}_s$, $\mathbf{I}_N\mathbf{y}_s = \mathbf{y}_s$, according to Lemma 4, we get

$$
\begin{aligned}
&(\mathbf{W}_1 \otimes \mathbf{H} + \mathbf{W}_1 \otimes \mathbf{I}_N)(\mathbf{x}_r \otimes \mathbf{y}_s) \\
&= (\mathbf{W}_1 \otimes \mathbf{H})(\mathbf{x}_r \otimes \mathbf{y}_s) + (\mathbf{W}_1 \otimes \mathbf{I}_N)(\mathbf{x}_r \otimes \mathbf{y}_s) \\
&= (\mathbf{W}_1\mathbf{x}_r) \otimes (\mathbf{H}\mathbf{y}_s) + (\mathbf{W}_1\mathbf{x}_r) \otimes (\mathbf{I}_N\mathbf{y}_s) \\
&= (\lambda_r\mathbf{x}_r) \otimes (\mu_s\mathbf{y}_s) + (\lambda_r\mathbf{x}_r) \otimes (\mathbf{y}_s) \\
&= (\lambda_r\mu_s + \lambda_r)(\mathbf{x}_r \otimes \mathbf{y}_s). \quad \square
\end{aligned}
$$

### *3.2. Case 2: $G_t$ shares the same $V_1$, and $\mathbf{W}_{t,t+1}$ is a full-rank matrix for $t \geq 1$*

We consider that all $V_t$ are composed of the same base nodes with $|V_t| = N$. However, $E_t$, $\mathbf{W}_t$ and $\mathbf{W}_{t,t+1}$ may be different for graph time slices, and $\mathbf{W}_{t,t+1}$, $t \geq 1$ are full-rank matrices. Suppose that the transformed graph $\mathcal{G}_T = (\mathcal{V}_T, \mathcal{E}_T)$ defined by the temporal-attention product satisfies the following assumption:

**Assumption (h′).** *The transformed graph $\mathcal{G}_T$ is composed of $G_t$ with the same base nodes. The weighted adjacency matrix $\mathcal{W}_T \in \mathbb{R}^{NT\times NT}$ satisfies* $\det(\mathbf{W}_{t,t+1}) \neq 0$.

Under this assumption, $\mathbf{W}_t$ and $\mathbf{W}_{t,t+1}$ are $N\times N$ matrices. Define a family of $N\times N$ matrix polynomials $\mathbf{P}_t(x)$. We call the zeros of $\mathbf{P}_t(x)$ the roots of the determinant of a matrix polynomial $\mathbf{P}_t(x)$, i.e., $\lambda$ is a zero of $\mathbf{P}_t(x)$ if $\det(\mathbf{P}_t(\lambda)) = 0$. The following theorem is very useful to calculate the eigenvalues and eigenvectors of the weighted adjacency matrix:

**Theorem 6.** *If the transformed graph $\mathcal{G}_T$ satisfies **Assumption** (h′), let $\mathbf{u}$ be an eigenvector of the weighted adjacency matrix $\mathcal{W}_T$. The eigenvector $\mathbf{u}$ corresponds to an eigenvalue $\lambda$ if and only if $\lambda$ is a zero of the matrix polynomial $\mathbf{P}_{T+1}(x)$, where $\mathbf{P}_{T+1}(x)$ satisfies the three-term recurrence relation:*

$$x\mathbf{P}_t(x) = \mathbf{W}'_{t-1,t}\mathbf{P}_{t-1}(x) + \mathbf{W}_t\mathbf{P}_t(x) + \mathbf{W}_{t,t+1}\mathbf{P}_{t+1}(x), \quad t = 1, 2, \cdots, T \tag{8}$$

*with $\mathbf{P}_0(x) = \mathbf{0}$, $\mathbf{P}_1(x) = \mathbf{I}_N$ and $\mathbf{W}_{0,1} = \mathbf{I}_N$. The eigenvector $\mathbf{u}$ has the form*

$$\mathbf{u} = \begin{pmatrix} \mathbf{P}_1(\lambda)\mathbf{y} \\ \vdots \\ \mathbf{P}_{T-1}(\lambda)\mathbf{y} \\ \mathbf{P}_T(\lambda)\mathbf{y} \end{pmatrix}, \tag{9}$$

*where $\mathbf{y} \in \mathbb{C}^N$ is a vector from the null space of the scalar matrix $\mathbf{P}_{T+1}(\lambda)$, i.e., the vector $\mathbf{y}$ satisfies $\mathbf{P}_{T+1}(\lambda)\mathbf{y} = 0$.*

The proof of the theorem can be directly obtained from the proof of Lemma 2.1 in [48]. When $T \to \infty$, the asymptotic behavior of eigenvalues has been shown in [27].

## 4. Signal processing on the dynamic graph

We consider a time-dependent graph signal $f$ defined on a discrete dynamic graph network, which is represented by a sequence of time-varying graphs $\{G_t = (V_t, E_t), t = 1, \cdots, T\}$. The definition of the graph signal is $f : \cup_t V_t \to \mathbb{R}$. By applying the temporal-attention product, we get the transformed graph $\mathcal{G}_T = (\mathcal{E}_T, \mathcal{V}_T)$ with node set $\mathcal{V}_T = \{\mathbf{v}_1, \cdots, \mathbf{v}_{N_T^{\#}}\}$, and edge set $\mathcal{E}_T$. Furthermore, we assume that $\mathcal{G}_T$ is connected and undirected. Then, we generalize graph Fourier transforms (GFT) and spectral graph wavelet transforms (SGWT) on the transformed graph $\mathcal{G}_T$.

### *4.1. Graph Fourier transform on the transformed graph*

Let $L^2(\mathcal{V}_T)$ be the collection of all real-valued square-summable functions on $\mathcal{V}_T$. $f \in L^2(\mathcal{V}_T)$ is a signal function defined on nodes of the transformed graph $\mathcal{G}_T$. Let $\mathcal{W}_T = (\mathbf{w}_{(\mathbf{v}_i,\mathbf{v}_j)})_{N_T^{\#} \times N_T^{\#}}$ be the weighted adjacency matrix of $\mathcal{G}_T$. The graph Laplacian matrix is defined as $\mathbf{L} = \mathbf{D} - \mathcal{W}_T$. $\mathbf{D} = (d_{ij})$ is a diagonal matrix with entries $d_{ii} = \sum_k \mathbf{w}_{(\mathbf{v}_i,\mathbf{v}_k)}$. For any $f \in L^2(\mathcal{V}_T)$, one can check that

$$\mathbf{L}f(\mathbf{v}_i) = \sum_{\mathbf{v}_j \in \mathcal{N}_{\mathbf{v}_i}^{1,T}} \mathbf{w}_{(\mathbf{v}_i,\mathbf{v}_j)}(f(\mathbf{v}_i) - f(\mathbf{v}_j)), \quad i, j = 1, \cdots, N_T^{\#}, \tag{10}$$

where $\mathcal{N}_{\mathbf{v}_i}^{1,T}$ denotes the spatio-temporal one-hop neighbor of $\mathbf{v}_i$ on $\mathcal{G}_T$.

Denote the non-negative, real-valued eigenvalues of $\mathbf{L}$ as $0 = \lambda_1 \le \lambda_2 \le \cdots \le \lambda_{N_T^{\#}}$, and the corresponding (normalized) eigenvectors are $\{\mathbf{u}_l, l = 1, \cdots, N_T^{\#}\}$. Eigenvalues and eigenvectors of graph Laplacian are closely related to almost all major invariants of a graph, and play an important role in understanding graphs in spectral graph theory. For a small eigenvalue $\lambda_l$, if an edge connects two nodes with nontrivial weight, the values of the eigenvector at those locations are more likely to be similar. On the other hand, eigenvectors associated with larger eigenvalues, oscillate more rapidly and are more likely to have dissimilar values on nearby neighbors.

The graph Laplacian eigenvectors and eigenvalues are analogous to the Fourier basis and frequencies [9,8]. The GFT $\widehat{f}$ of any signal $f \in L^2(\mathcal{V}_T)$ on the nodes of $\mathcal{G}_T$ can be defined as:

$$\widehat{f}(\lambda_l) = \sum_{i=1}^{N_T^{\#}} \mathbf{u}_l(\mathbf{v}_i) f(\mathbf{v}_i), \quad l = 1, \cdots, N_T^{\#}. \tag{11}$$

The high-frequency Fourier coefficients indicate that a signal varies abruptly in some regions of the graph, whereas low-frequency Fourier coefficients indicate smooth signal variation in some instances.

The computation of the Fourier coefficients from (11) requires a cost of $O({N_T^{\#}}^3)$ operations. Based on the techniques in [49], we can give a theorem to reduce the computational complexity of the graph Fourier transform under Assumption (h′).

**Theorem 7.** *Suppose that the transformed graph $\mathcal{G}_T$ satisfies* ***Assumption (h′)****. Let $f_T$ be the signal defined on $\mathcal{V}_T$. The diagonal matrix $\mathbf{D}$ can be expressed as a block diagonal matrix $\mathbf{D} = \mathrm{diag}(\mathbf{D}_1, \cdots, \mathbf{D}_\mathrm{T})$ with each entry $\mathbf{D}_t \in \mathbb{R}^{N \times N}$. A family of $N \times N$ matrix polynomials $\mathbf{P}_t(x)$ satisfies*

$$x\mathbf{P}_t(x) = -\mathbf{W}'_{t-1,t}\mathbf{P}_{t-1}(x) + (\mathbf{D}_t - \mathbf{W}_t)\mathbf{P}_t(x) - \mathbf{W}_{t,t+1}\mathbf{P}_{t+1}(x), \quad t = 1, 2, \cdots, T \tag{12}$$

*with $\mathbf{P}_0(x) = \mathbf{0}$, $\mathbf{P}_1(x) = \mathbf{I}_N$ and $\mathbf{W}_{t-1,t} = \mathbf{I}_N$. Let $\mathbf{Y}_m$ be an $N \times a_m$ matrix with columns given by basis vectors for the null space of $\mathbf{P}_{T+1}(\lambda_m)$, where $\lambda_m$ is a zero of $\mathbf{P}_{T+1}(x)$ with multiplicity $a_m$. Then*

*(i). $\lambda_m$ is the eigenvalue of Laplace matrix $\mathbf{L}$. The columns of the matrix*

$$\begin{pmatrix} \mathbf{P}_1(\lambda_m)\mathbf{Y}_m \\ \vdots \\ \mathbf{P}_T(\lambda_m)\mathbf{Y}_m \end{pmatrix} \tag{13}$$

*are linearly independent eigenvectors corresponding to the eigenvalue* $\lambda_m$, $1 \leq m \leq m_0$, *where* $m_0$ *is the number of distinct eigenvalues.*

*(ii). The graph Fourier transform of* $f_T$ *is*

$$\widehat{f_T} = \mathbf{U}' f_T,$$

*where*

$$\mathbf{U} = \begin{pmatrix} \mathbf{P}_1(\lambda_1)\mathbf{Y}_1 & \cdots & \mathbf{P}_1(\lambda_{m_0})\mathbf{Y}_{m_0} \\ \vdots & & \vdots \\ \mathbf{P}_T(\lambda_1)\mathbf{Y}_1 & \cdots & \mathbf{P}_T(\lambda_{m_0})\mathbf{Y}_{m_0} \end{pmatrix}. \tag{14}$$

**Proof.** (i). Since $\lambda_m$ is a zero of $\mathbf{P}_{T+1}(x)$ and $\mathbf{P}_{T+1}(x)$ satisfies (12), according to Theorem 6, $\lambda_m$ is the eigenvalue of Laplace matrix $\mathbf{L}$. Let $\mathbf{y}_k$ be the column in $\mathbf{Y}_m$, $k = 1, \cdots, a_m$. By definition, $\mathbf{y}_k$ is the basis vector for the null space of $\mathbf{P}_{T+1}(\lambda_m)$. We have that $\mathbf{y}_k$ with $k = 1, \cdots, a_m$ is linearly independent. Since $\mathbf{P}_1(\lambda_m) = \mathbf{I}_N$, the columns of (13) are linearly independent eigenvectors corresponding to the eigenvalue $\lambda_m$.

(ii). Since the graph Laplacian matrix $\mathbf{L}$ is a symmetric matrix, it can be diagonalized. According to (12),

$$\mathbf{L}\begin{pmatrix} \mathbf{P}_1(\lambda_m)\mathbf{Y}_m \\ \vdots \\ \mathbf{P}_T(\lambda_m)\mathbf{Y}_m \end{pmatrix} = \begin{pmatrix} \lambda_m \mathbf{P}_1(\lambda_m)\mathbf{Y}_m \\ \vdots \\ \lambda_m \mathbf{P}_T(\lambda_m)\mathbf{Y}_m + \mathbf{W}_{T,T+1}\mathbf{P}_{T+1}(\lambda_m)\mathbf{Y}_m \end{pmatrix} = \begin{pmatrix} \mathbf{P}_1(\lambda_m)\mathbf{Y}_m \\ \vdots \\ \mathbf{P}_T(\lambda_m)\mathbf{Y}_m \end{pmatrix} \lambda_m \mathbf{I}_{a_m}. \tag{15}$$

Hence, for $\mathbf{U}$ in (14), we obtain

$$\mathbf{L}\mathbf{U} = \mathbf{U}\begin{pmatrix} \lambda_1 \mathbf{I}_{a_1} & & \\ & \ddots & \\ & & \lambda_{m_0}\mathbf{I}_{a_{m_0}} \end{pmatrix}. \tag{16}$$

This decomposition is equal to the eigendecomposition, which confirms that $\mathbf{U}$ is an eigenvector matrix for $\mathbf{L}$. This proof borrows the idea of [49]. □

Direct computation of the eigenvectors requires the cost of $O(N^3T^3)$ under Assumption (h′). Instead, we can calculate the bases of the null spaces of $\mathbf{P}_{T+1}(\lambda_m)$, which requires only $O(N^2m_0)$ operations, and compute $NT$ products of $N \times N$ matrices with vectors of length $N$, which requires $O(N^3T)$ operations. The total operations required are $O(N^2m_0) + O(N^3T) = O(N^3T)$. It can reduce the calculation cost for long-term dynamic research of graphs of appropriate size.

For larger graphs and more general cases, such as when the number of nodes on each graph time slice varies, we can calculate the approximate eigenvalues and eigenvectors in parallel using the divide-and-conquer method [41,42,53], or apply the twisted block factorizations method [43,44].

### *4.2. Spectral graph wavelet transform on the transformed graph*

The SGWT is proposed in [9,8] to analyze the local properties of the signal on the graph by first introducing a graph spectral filter dictionary. $\{\hat{g}_m | m = 1, 2, \cdots, M\}$ represents a collection of graph spectral filters, and $M$ denotes the number of graph spectral filters in the dictionary. Let $\delta_{\mathbf{v}}$ be the one-hot vector at $\mathbf{v}$ for any $\mathbf{v} \in \mathcal{V}_T$. The spectral graph wavelet indexed at $(m, \mathbf{v})$ is then defined as

$$\psi_{m,\mathbf{v}} := \mathbf{U}\mathbf{D}_{\hat{g},m}\mathbf{U}'\delta_{\mathbf{v}},$$

with scale $m \in \mathbb{R}^+$ and centered on node $\mathbf{v}$. $\mathbf{U}$ is the unitary matrix, and the columns are given by the eigenvectors $\mathbf{u}_l$, $l = 1, \cdots, N_T^{\#}$.

$$\mathbf{D}_{\hat{g},m} = \mathrm{diag}(\hat{\mathrm{g}}_{\mathrm{m}}(\lambda_1), \cdots, \hat{\mathrm{g}}_{\mathrm{m}}(\lambda_{\mathrm{N}_{\mathrm{T}}^{\#}}))$$

is a diagonal matrix.

The SGWT of $f$ can be defined as the wavelet coefficient at index $(m, \mathbf{v})$, which can be calculated by:

$$W_f(m, \mathbf{v}) := \langle f, \psi_{m,\mathbf{v}} \rangle = \sum_{l=1}^{N_T^{\#}} \hat{g}_m(\lambda_l) \widehat{f}(\lambda_l) \mathbf{u}_l(\mathbf{v}). \tag{17}$$

This is, indeed, a generalized Fourier transform with kernel $\hat{g}_m(\lambda_l)$. Spectral graph wavelets, like conventional wavelets, are localized in both frequency and time. The low-frequency wavelet coefficients (corresponding to small $m$ values) in node $\mathbf{v}$ are greater than the high-frequency coefficients (corresponding to large $m$ values), indicating a smoother signal fluctuation. In contrast, larger coefficients appear in high frequencies in node $\mathbf{v}$, indicating that the signal oscillates more abruptly on and around this node.

In this paper, we choose a wavelet dictionary proposed in [8] – the spectral graph wavelet (SGW) dictionary. The dictionary is defined as $\hat{g}_1(\lambda) = \hat{h}(\lambda)$, and $\hat{g}_m(\lambda) = \hat{g}(s_{M-m+2}\lambda)$, for $m = 2, \cdots, M$. In this paper, we set $M = 8$.

$$\hat{g}(\lambda) = \begin{cases} \lambda^2, & 0 \le \lambda < 1 \\ -5 + 11\lambda - 6\lambda^2 + \lambda^3, & 1 \le \lambda \le 2 \\ 4\lambda^{-2}, & 2 < \lambda \end{cases} \tag{18}$$

is a bandpass filter defined on the Fourier domain. The stretching scales $s_2, s_3, \cdots, s_M$ are sampled logarithmically between $s_2 = 1/\lambda_{N_T^{\#}}$ and $s_M = 40/\lambda_{N_T^{\#}}$. To represent the low frequency component of the signal $f$, a scaling function acts as a low-pass filter:

$$\hat{h}(\lambda) = \gamma \exp\Big( - \Big((10\lambda)/(0.3\lambda_{N_T^{\#}})\Big)^4 \Big). \tag{19}$$

The parameter $\gamma$ is chosen such that $\hat{h}(0)$ is equal to the maximum value of $\hat{g}$.

(17) requires eigendecomposition and has computational complexity of $O(N_T^{\#^3})$. A fast spectral graph wavelet transform based on Chebyshev polynomials approximation was proposed in [8]. It is shown that (17) can be approximated by

$$W_f(m, \mathbf{v}) \approx \Big( c_{m,0} f/2 + \sum_{k=1}^{K_m} c_{m,k} \overline{T}_k(L) f \Big)(\mathbf{v}), \tag{20}$$

where

$$c_{m,k} = (2/\pi) \int_0^{\pi} \cos(k\theta) \hat{g}_m(\lambda_{N_T^{\#}}(\cos(\theta) + 1)/2) \mathrm{d}\theta,$$

and $K_m$ represents the number of truncating terms. In this paper, we set $K_m = 40$. Let $\overline{T}_k$ be the shifted Chebyshev polynomials with the domain of $[0, \lambda_{N_T^\#}]$. $\overline{T}_k(\mathbf{L})$ satisfies the recursive formula

$$\overline{T}_k(\mathbf{L})f = (4/\lambda_{N_T^\#})\left(\mathbf{L} - (\lambda_{N_T^\#}/2)\mathbf{I}_{N_T^\#}\right)\overline{T}_{k-1}(\mathbf{L})f - \overline{T}_{k-2}(\mathbf{L})f,$$

for $2 \le k \le K_m$, with initial conditions

$$\overline{T}_0(\mathbf{L})f = f,\ \ \overline{T}_1(\mathbf{L})f = (2/\lambda_{N_T^\#})\left(\mathbf{L} - (\lambda_{N_T^\#}/2)\mathbf{I}_{N_T^\#}\right)f,$$

where $\mathbf{L}$ is the graph Laplacian and $f$ is the signal. Therefore, we only need to estimate the maximum eigenvalue when performing SGWT. The computational cost to approximate the wavelet coefficients is order $O(K_m|\mathcal{E}| + K_m N_T^\#)$. If the transformed graph satisfies Assumption (h′) and the calculation cost permits, we can compute the wavelet coefficients specified in (17) based on Theorem 7. Additionally, using the recurrence formula, it is beneficial to study the evolution of its wavelet coefficients when the time slice is expanded. In particular, the weighted adjacency matrix of the transformed graph is recursive, we only need to calculate $\mathbf{P}_{T+1}$ for the new time slice $G_{T+1}$.

**Remark 1.** Graph signal processing is usually considered under an undirected graph framework, since $\mathbf{L}$ may not have a complete set of eigenvectors for a directed graph. For a directed graph, we can consider the extended Laplacian denoted by $\mathbf{L}_{sym} = (\mathbf{L} + \mathbf{L}')/2$. Then $\mathbf{L}_{sym}$ is a semi-positive symmetric matrix, and its eigenvectors form a set of orthonormal bases in $L^2(\mathcal{V}_T)$.

## 5. Classification method based on SGWT coefficients

SGWT coefficients contain rich information about the graph signal; however, it remains a big challenge to interpret them properly for non-experts. In this section, we will introduce the classification and visualization methods based on SGWT coefficients, mainly based on the literature [23,54].

### *5.1. Node classification using SGWT coefficients*

Through (17), we obtain the wavelet coefficients: $[W_f(1, \mathbf{v}), \cdots, W_f(M, \mathbf{v})]$, where $M$ denotes the number of graph spectral filters in the dictionary. Next, we use wavelet coefficients to classify nodes:

Firstly, a robust scaler transform introduced in RobustScaler of the scikit-learn library [55] is applied by

$$S(m, \mathbf{v}) = |W_f(m, \mathbf{v})|/\mathrm{IQR}(m), \quad m = 1, \cdots, M, \tag{21}$$

where $\mathrm{IQR}(m)$ is the Inter-Quartile Range (between the first and the third quartile) of $|W_f(m, \mathbf{v})|$. This transform ensures that wavelet coefficients are on the same scale.

Secondly, to make it easier to compare the torque values between nodes, each coefficient is normalized using a logarithmic normalization [56]

$$\overline{W}_f(m, \mathbf{v}) = \ln(1 + S(m, \mathbf{v}))\,/\,\ln\left(1 + \max_{\mathbf{v}} S(m, \mathbf{v})\right), \quad m = 1, \cdots, M, \tag{22}$$

to the range [0, 1]. The normalized wavelet coefficients can be represented as a vector:

$$\overline{W}_f(\mathbf{v}) = [\overline{W}_f(1, \mathbf{v}), \cdots, \overline{W}_f(M, \mathbf{v})].$$

Thirdly, the torque function $\varphi : \mathcal{V}_T \to \mathbb{R}$ is defined as [23]:

$$\varphi(\mathbf{v}) = \overline{W}_f(\mathbf{v}) \cdot Z, \tag{23}$$

| Node class | $\mathcal{V}_T^1$ | $\mathcal{V}_T^2$ | $\mathcal{V}_T^3$ | $\mathcal{V}_T^4$ | $\mathcal{V}_T^5$ |
|---|---|---|---|---|---|
| Signal frequency | low | mid-low | average | mid-high | high |
| Symbol | ■ | 🟦 | ☆ | 🟧 | 🟥 |

**Fig. 3.** Node classification. Nodes in $\mathcal{V}_T$ are divided into five classes $\mathcal{V}_T^1$, $\mathcal{V}_T^2$, $\mathcal{V}_T^3$, $\mathcal{V}_T^4$, and $\mathcal{V}_T^5$, representing the spatio-temporal changes of the signal (in terms of the SGWT coefficients) as low frequency, mid-low frequency, average, mid-high frequency, and high frequency, respectively.

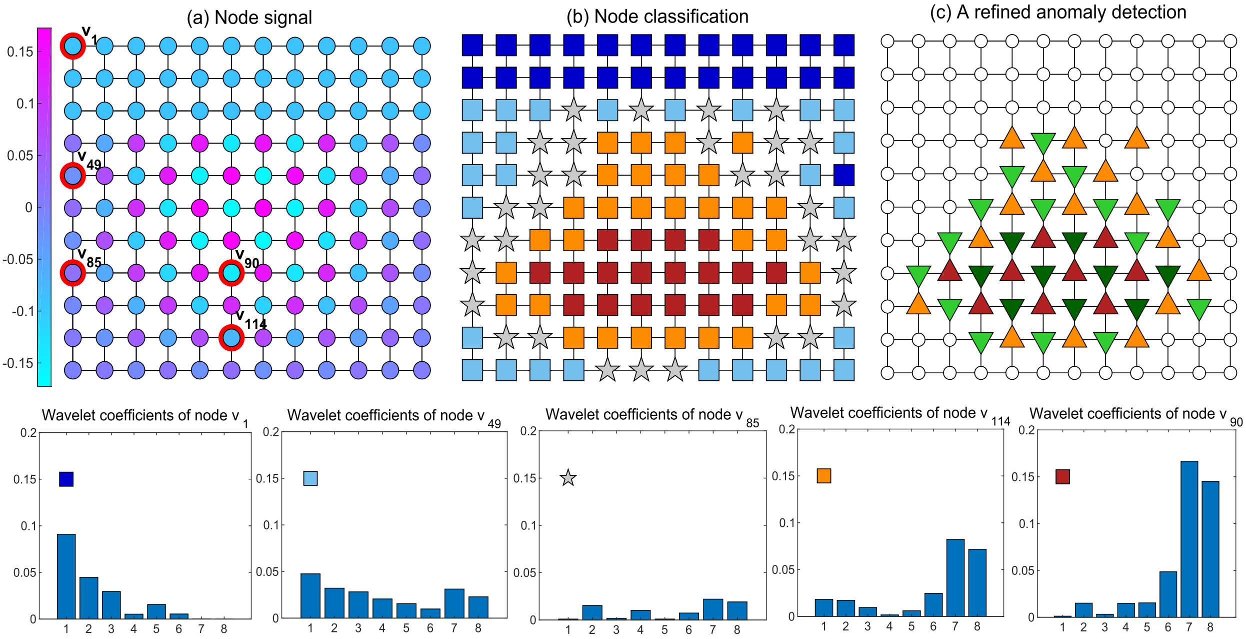


**Fig. 4.** Node signals, node classification, and refined node classification based on SGWT. The refined classification method will be introduced in section 5.2 with the hyperparameters $\zeta_1 = 1$ and $\zeta_2 = 1$. The bottom shows the wavelet coefficients of some nodes in five classes.

where $Z = [-M/2, \cdots, -1, 1, \cdots, M/2]$ is a (signed) weight vector and '$\cdot$' denotes inner product. Clearly, for any $\mathbf{v} \in \mathcal{V}_T$, the torque value $\varphi(\mathbf{v})$ is a weighted sum of the normalized SGWT coefficients. The higher the torque value, the more severe the signal change. The lower the torque value, the smoother the signal.

Finally, let $\varphi_{\min} = \min\limits_{\mathbf{v}\in\mathcal{V}_T} \varphi(\mathbf{v})$, $\varphi_{\max} = \max\limits_{\mathbf{v}\in\mathcal{V}_T} \varphi(\mathbf{v})$. Similar to [54], we define a classification score as $\sigma : \mathcal{V}_T \to \{1, 2, 3, 4, 5\}$

$$\sigma(\mathbf{v}) = [5(\varphi(\mathbf{v}) - \varphi_{\min})/(\varphi_{\max} - \varphi_{\min})], \tag{24}$$

where $[\cdot]$ is the integer function. Thus $\mathcal{V}_T$ can be divided into five classes, as shown in Fig. 3.

We give 11 × 12 square lattices to illustrate that this classification method identifies the change of signal well. The signal on it is defined as the combination of the low and high frequency eigenvectors: $f(\mathbf{v}_i) = \mathbf{u}_1(\mathbf{v}_i)$, if $1 \le i \le 36$; $f(\mathbf{v}_i) = \mathbf{u}_{132}(\mathbf{v}_i)$, if $37 \le i \le 132$. Fig. 4 (a) describes the signal value on the graph, where the color associated with the graph nodes encodes the graph signal. Fig. 4 (b) demonstrates the node classification. When the signal is significantly different from the neighbor node, it is classified as $\mathcal{V}_T^5$, and when the signal is similar to the neighbor node, it is classified as $\mathcal{V}_T^1$. The bottom shows the wavelet coefficients of some nodes in five classes.

### 5.2. A refined node classification for anomaly detection

Effective anomaly detection could give early indications of danger or find interesting phenomena. This is necessary for disease outbreak detection, genetic network analysis, activity monitoring in social networks,

| Anomaly class | $\mathcal{C}_{-2}$ | $\mathcal{C}_{-1}$ | $\mathcal{C}_{0}$ | $\mathcal{C}_{1}$ | $\mathcal{C}_{2}$ |
|---|---|---|---|---|---|
| Symbol | ▼ | ▼ | None | ▲ | ▲ |

**Fig. 5.** Refined anomaly detection on $\mathcal{V}_T^4 \cup \mathcal{V}_T^5$ by a-scores.

environmental monitoring, malware detection, spam filtering, etc [57,58]. SGWT is a powerful tool to detect the spatio-temporal anomaly of the signal with the help of the transformed graph. We say the nodes in $\mathcal{V}_T^4 \cup \mathcal{V}_T^5$ are anomaly nodes. However, as shown in Fig. 4, for example, there are completely different types of nodes in $\mathcal{V}_T^5$ by examining the distribution of the signal in more detail. As we can see for node $\mathbf{v}_{90}$, the signal value is significantly lower than that of its neighbors; while for the one-hop neighbor of $\mathbf{v}_{90}$, the signal value is higher than the surrounding values.

A refined node classification method based on [54] is used to identify these two different types of anomalies for nodes in $\mathcal{V}_T^4 \cup \mathcal{V}_T^5$. Given a graph signal $f \in L^2(\mathcal{V}_T)$, a new metric $\vartheta : V_t \to \mathbb{R}$ is defined as

$$\vartheta[(v,t)] = \max\{f[(v,t)] - \zeta_1 \bar{f}[(v,t)], 0\} + \min\{f[(v,t)] - \zeta_2 \bar{f}[(v,t)], 0\}, \tag{25}$$

where

$$\bar{f}[(v,t)] = (1/|N^1_{(v,t)}|) \sum_{(w,t)\in N^1_{(v,t)}} f[(w,t)], \tag{26}$$

$N^1_{(v,t)}$ is the one-hop neighbor of $(v,t) \in V_t$ on the $t$-th time slice $G_t$. $\zeta_i$ is a hyperparameter, which is used to further subdivide the anomaly class.

Next, the anomaly score (a-score) function is $\chi : \mathcal{V}_T \to \{-2,-1,0,1,2\}$,

$$\chi[(v,t)] = \begin{cases} 0 & (v,t) \in \mathcal{V}_T^i, i = 1,2,3, \\ (i-3)\cdot \operatorname{sign}(\vartheta[(v,t)]) & (v,t) \in \mathcal{V}_T^i, i = 4,5. \end{cases} \tag{27}$$

Thus $\mathcal{V}_T^4$ and $\mathcal{V}_T^5$ are further divided into two sub-classes. According to the value of the a-score, we thus have a new classification of the nodes into the level sets of the a-score function as $\mathcal{V} = \mathcal{C}_{-2} \cup \mathcal{C}_{-1} \cup \mathcal{C}_0 \cup \mathcal{C}_1 \cup \mathcal{C}_2$, with $\mathcal{C}_i = \{(v,t)|i = \chi[(v,t)]\}$. This is also shown in Fig. 5, where we use different colors to represent these anomaly classes. The a-score depends on both the spatial and temporal relationship of graph signals. Using the a-score, we can obtain Fig. 4 (c), which shows the internal differences in $\mathcal{V}_T^4 \cup \mathcal{V}_T^5$ in great detail.

### *5.3. Graph classification based on SGWT coefficients*

A graph classification method is presented to classify each graph time slice $\{G_t, t = 1, \cdots, T\}$ on the transformed graph $\mathcal{G}_T$. The distribution of the $i$-th class $\mathcal{V}_T^i$ on time slice $G_t$ is calculated by

$$\rho_t^i = (1/N_t) \sum_{(v,t)\in V_t} \mathbb{I}_{\mathcal{V}_T^i}(v,t), \quad i = 1,2,\cdots,5, \tag{28}$$

where $\mathbb{I}_{\mathcal{V}_T^i}$ is the indicator function of the $i$-th class $\mathcal{V}_T^i$. We use the method in [23] to classify the graph time slices $\{G_t, t = 1, \cdots, T\}$. The graph classification function of $G_t$ is defined as:

$$\sigma_t = \operatorname*{arg\,max}_{i\in\{1,2,3,4,5\}} (\rho_t^i/\rho_{\max}^i), \tag{29}$$

where $\rho_{\max}^i = \max\{\rho_1^i, \cdots, \rho_T^i\}$. We thus classify the $t$-th graph time slice as $\mathcal{V}_T^{\sigma_t}$, which carries the most weight in time slice $G_t$. The graph time slice belonging to $\mathcal{V}_T^4 \bigcup \mathcal{V}_T^5$ is known as the anomaly graph time slice.

## 6. Spectral graph wavelet visual analysis on dynamic graphs

In this section, we implement SGWT on two real datasets. By classifying nodes based on wavelet coefficients, it is shown that implementing SGWT on the transformed graph can catch abnormal events based on signal changes and accurately find interesting key information. We use Matlab to perform fast SGWT and visualization. In the case study 1, we use two types of signals, and calculating the wavelet coefficients takes 0.762323 and 0.737458 seconds, respectively. In the case study 2, it costs 0.029278 seconds to compute wavelet coefficients. Also, we train the graph attention neural network using Pytorch packages on a single NVIDIA GeForce RTX 3090. It costs 5017.4465 seconds.

### 6.1. Case study 1: 2009 SFHH conference in Nice

We choose a dynamic network with a time-dependent graph topology to test the effectiveness of our proposed method for dynamic graph signal processing. We use the dataset provided in [59], which describes the face-to-face interactions of 405 participants at the 2009 SFHH conference in Nice, France (June 4-5, 2009). The data collection time was from 9:00 am to 9:00 pm on the first day, and from 8:30 am to 4:30 pm on the second day. The original data provided the pair-wise contact information among participants at every 20-second interval during the two-day periods.

#### 6.1.1. Construction of the dynamic graph

We first construct the discrete graph time slices $\{G_t = (V_t, E_t),\, t = 1, \cdots, T\}$ with $T = 22$, where the time interval is one hour. Because this dataset was recorded at a conference, the one-hour time period allows us to better identify the size of the sub-communities where people interacted with one another. Here we omitted the time slices at night when there is no communication. Note that someone may not communicate with others in some time slices, and $\mid V_t \mid$ and $E_t$ change with time. The link weight $w_{(v,w,t)}$ is proportional to the length of the conversation and is the conversation count between $(v,t)$ and $(w,t)$ every 20 seconds within an hour. Thus we get symmetric, undirected graphs $\{G_t, t = 1, \cdots, T\}$.

In this section, we take two types of graph signal functions to analyze the conference data. Let

$$f_1[(v,t)] = \sum_{(w,t)\in N^1_{(v,t)}} w_{(v,w,t)},$$

where $f_1[(v,t)])$ represents the total conversation counts of participant $v$ during the $t$-th time slice. Let

$$f_2[(v,t)] = \sum_{(w,t)\in V_t} \mathbb{I}_{E_t}(e_{(v,w,t)}),$$

where $f_2[(v,t)]$ represents how many people $v$ contacted at the $t$-th time slice. Next, we construct the transformed graph $\mathcal{G}_T$ using the temporal-attention product. We also link the time slice 21:00-22:00, June 4 to the time slice 8:30-9:30, June 5 by temporal-attention product. Suppose $\mathbf{w}_{((v,t),(w,t+1))}$ inherits from $w_{(v,w,t)}$ following the temporal evolution procedure, i.e., $\mathbf{w}_{((v,t),(w,t+1))} := w_{(v,w,t)}$, $\mathbf{w}_{((v,t),(w,t))} := w_{(v,w,t)}$. There are some small groups (containing only 2-3 nodes) in which the participants only discuss within the group and do not communicate with other groups. When we perform the analysis, we do not consider these groups. So the transformed graph we get is a connected graph with $\mid \mathcal{V}_T \mid= 5008$. Then, we compute the graph wavelet coefficients of the graph signal $f_1[(v,t)]$ and $f_2[(v,t)]$, respectively.

#### 6.1.2. Overall visual analysis

We classify each graph time slice $\{G_t, t = 1, \cdots, T\}$ on the transformed graph $\mathcal{G}_T$ by the graph classification method designed in section 5. Firstly, we assign each node $(v,t)$ at $t$-th time slice to one of the

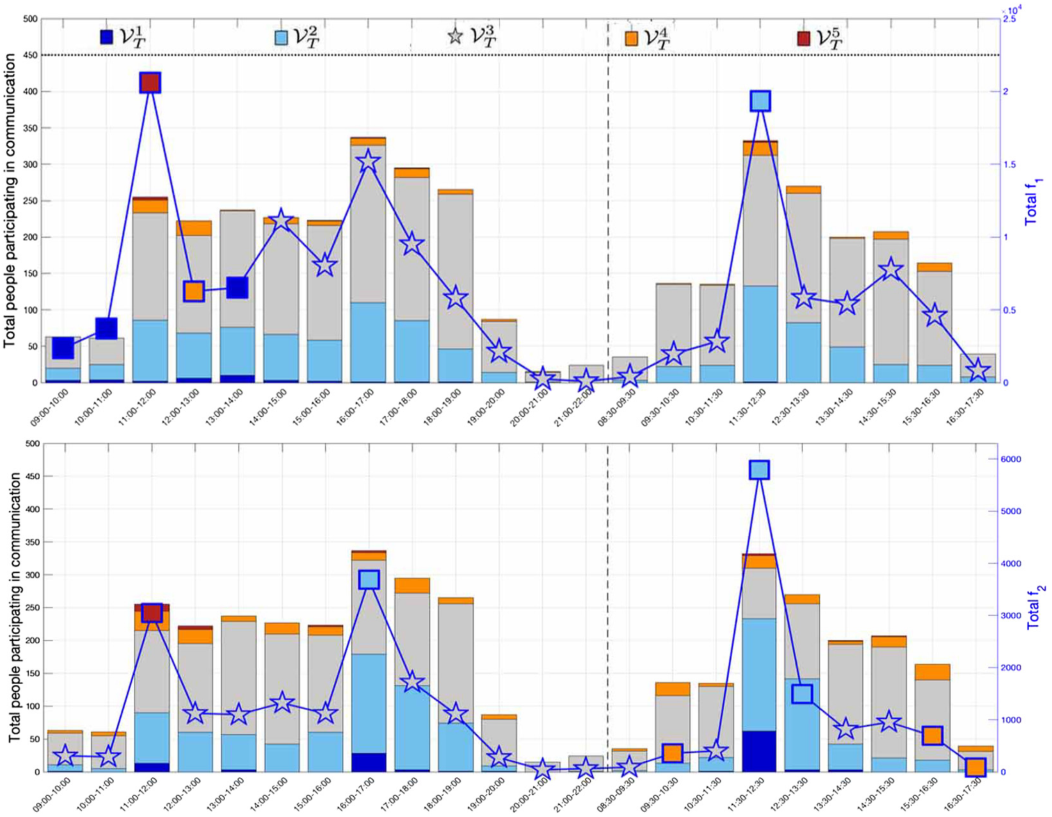


**Fig. 6.** The bar graph depicts the node classification distribution $\rho_t^i$ in (28). The line chart (in blue) depicts the graph classification for each graph time slice and the sum of signals. The subfigures on top and bottom are based on the graph signal $f_1$ and $f_2$, respectively. The dotted line marks the start of the second-day meeting.

classes $\{\mathcal{V}_T^i, i = 1, \cdots, 5\}$ by the node classification function $\sigma[(v, t)]$ defined as in (24). Secondly, we show the corresponding probability distribution $\rho_t^i$ defined in (28), see the bar graph of Fig. 6. The color for the probability distribution $\rho_t^i$ matches that of $\mathcal{V}_T^i$. Finally, using the graph classification function $\sigma_t$ defined in (29), we can assign each graph time slice a class $\mathcal{V}_T^{\sigma_t}$ with $\sigma_t \in \{1, 2, 3, 4, 5\}$. This allows us to better understand how active the participants are in the meeting. The sum of $f_1$ and the sum of $f_2$ on each time slice, are plotted as time series in the line chart of Fig. 6, respectively. Graph time slices are denoted by different shapes and colors to indicate the class from $\{\mathcal{V}_T^{\sigma_t}, t = 1, \cdots, 5\}$, representing which spatio-temporal changes (low frequency, mid-low frequency, average, mid-high frequency, and high frequency) of node signals in the time slice account for the greater proportion.

On June 4, 11:00-12:00, it can be seen from bar graphs Fig. 6 that more than 250 participants had face-to-face communication. Both graph signals reach their peaks, and the graph time slice $G_3$ is classified as a high frequency class. This means high-frequency signals in this time slice accounted for the most significant proportion of the entire 22 hours-period. It can be inferred that there exists a lunch break and many people start a conversation. It was the first break of the meeting, and most people were very active, greeting each other out of politeness or making new friends. People with a large social circle will contact multiple people at this time. The sizes of the social circle of participants are also different. At the same time, the total time for one participant to communicate is also more than other time slices. These factors explain well that $G_3$ belongs to $\mathcal{V}_T^5$ in $\mathcal{G}_T$.

As an interesting phenomenon, we can see that on June 5, 12:30-13:30 ($G_{17}$), nearly 330 people had face-to-face communication, which counts for another peak time of the signals. However, $G_{17}$ is identified as a mid-low frequency class (in blue), contrary to $G_3$, using the graph classification method. This should

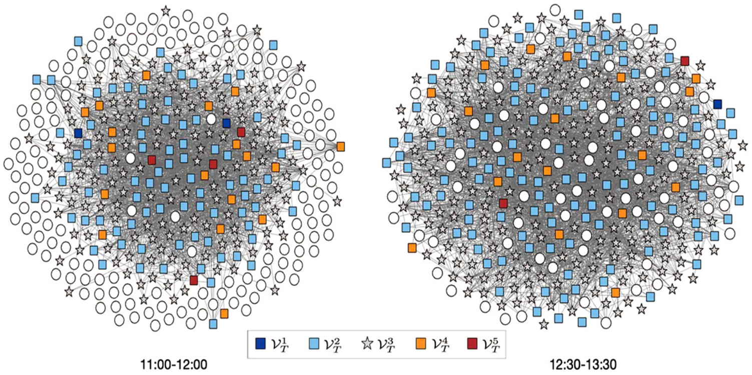


**Fig. 7.** Node classification on $G_3$ (left) and $G_{17}$ (right) using the graph signal $f_1$. One can see clearly the graph signal on $G_{17}$ is almost uniform and the low frequency accounts for more, thus should be classified as a normal subgraph; while $G_3$ is definitely an anomaly one.

be the lunchtime of the second day, and we can read from the plot that, during this hour, most participants had the same social profile as their social objects. It can be speculated that people in the same social circle communicate with each other, and there are few brief greetings between different social circles. This is also in line with reality, because on the first-day lunch everyone will greet people they know or not out of courtesy, and there will be brief exchanges between different social circles. But before leaving, most participants prefer to have rather long conversations with friends. Therefore, it is not difficult to explain why the classification method identifies this hour as a mid-low frequency class, instead of a high frequency class, compared to the first lunchtime, using the graph signals.

To better assist in visualizations, we also plot graphs $G_3$ and $G_{17}$, see Fig. 7, together with the node classifications on both graph time slices. It is consistent with the graph classifications.

### *6.1.3. Ranking participants by popularity*

According to the a-score defined in (27), we can find the most popular participants at the conference, using the two graph signal functions. Indeed, based on these two graph signals, we further define a comprehensive anomaly score to analyze participants' popularity:

$$\bar{\chi}'(v) := (1/T)\sum_{t=1}^{T} \chi_{f_1}[(v,t)] \cdot \chi_{f_2}[(v,t)], \tag{30}$$

where $\chi_{f_1}[(v,t)] > 0$ and $\chi_{f_2}[(v,t)] > 0$ are the a-scores defined in (27) with the hyperparameter $\zeta_1 = \zeta_2 = 1$. We can easily rank the popularity of participants through the comprehensive anomaly score. The larger the value, the more popular it is.

Fig. 8 shows the top five most popular (active) participants ranked using a comprehensive anomaly score (30) based on graph signals $f_1$ and $f_2$. The first column of Fig. 8 is the score $\bar{\chi}'(v)$ defined in (30). The view also displays the a-scores (27) of participants on different time slices. To distinguish the two signals, we use circles to represent the a-score based on signal $f_1$, and triangles to represent the a-score based on signal $f_2$. According to (30), a score is recorded when the two signal markers in Fig. 8 overlap, indicating that the participant's talk time and the number of people he talks to are both greater than the average of his neighbors at this time slice. That is, the participant is more popular if more people are willing to spend more time communicating with him compared to others. As we can see that participants $v_{84}, v_{21}$ and $v_{64}$ were identified as the most popular people during the first daytime; while $v_{68}$ and $v_{21}$ were mostly active during the dinnertime of the first day. Participant $v_{76}$ was the most popular one during the entire conference, as he was very active for almost all the recorded time.

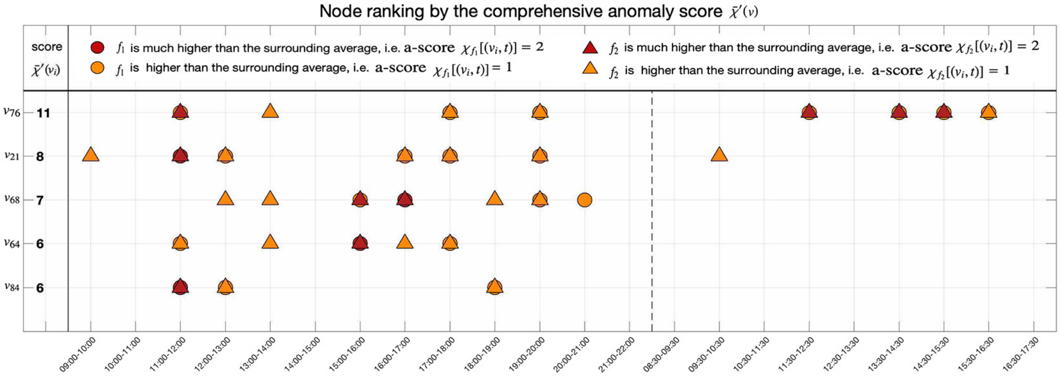


**Fig. 8.** Node ranking. The circle corresponds to the signal $f_1$, and the triangle corresponds to the signal $f_2$. The dotted line marks the start of the second-day meeting. The first column is the score $\bar{\chi}'(v)$ defined in (30). $\chi_{f_i}[(v,t)]$ is the a-score defined in (27).

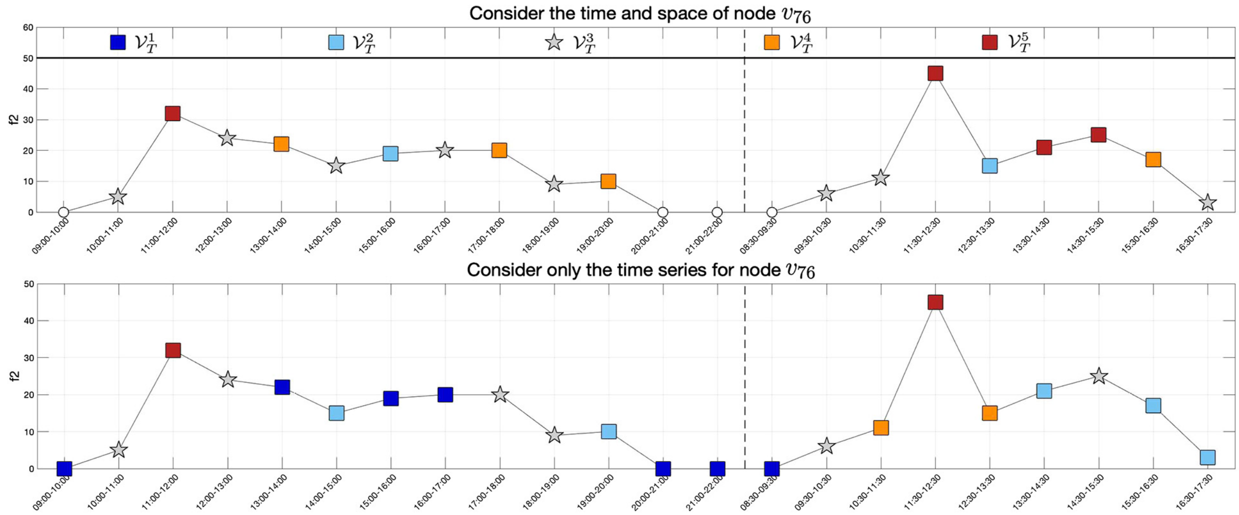


**Fig. 9.** Top: node classifications for participant $v_{76}$ on the transformed graph. Bottom: node classifications for participant $v_{76}$ only using a temporal series graph. The signal is $f_2$. The dotted line marks the start of the second-day meeting.

We further provide a detailed graph signal analysis of participant $v_{76}$ in Fig. 9. We visualize the temporal activity of the participants in each time slice, using the node classifications by graph signal $f_2$. Interestingly, the classification by only the time-series graph of participant $v_{76}$ (bottom) is rather different from the classification on the transformed graph (top). This also explains that it is not sufficient to perform the node classifications and anomaly detections by only the time-series.

To demonstrate that the dynamic model connected by temporal-attention product can better enable SGWT to capture the changes of signal dynamic mode, we supplement the node rankings of dynamic models connected by Cartesian product and strong product, respectively, in the Appendix.

### *6.2. Case study 2: COVID-19 data analysis among the Group of Twenty*

The World Health Organization (WHO) on March 11, 2020, declared the novel coronavirus (COVID-19) outbreak a global pandemic. There are many papers studying the spread of COVID-19 from different perspectives, which include how national mitigation measures will affect the process of the COVID-19 pandemic [60], the comparison of pandemic situations among different countries [61,62], and the impact on social life during the pandemic period [63–66].

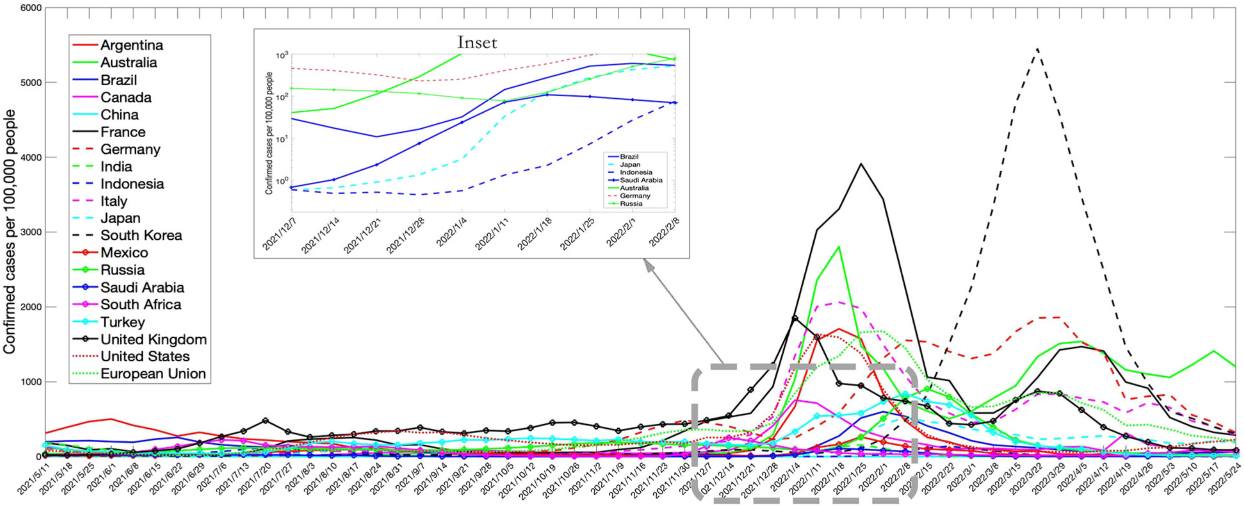


**Fig. 10.** Time series of the confirmed COVID-19 cases per 100,000 people every week. The inset is the zoom-in view of some country curves from Dec. 7, 2021 to Feb. 8, 2022, where the y-axis is log-scale.

To understand the roles of individual countries played in the pandemic spreading, we choose to analyze COVID-19 data among the Group of Twenty (G20): Argentina, Australia, Brazil, Canada, China, France, Germany, India, Indonesia, Italy, Japan, South Korea, Mexico, Russia, Saudi Arabia, South Africa, Turkey, the United Kingdom, the United States, and the European Union. The data is taken from the official website of the World Health Organization.[2] The original data collected daily new cases in each country. Considering some countries that do not update their data daily, we summed the data every seven days to analyze the weekly new cases. The data collection period spans from May 5, 2021 to May 24, 2022. For simplicity, we used the last date of each week as the label for that particular week in the subsequent visualization analysis. The signals are the multivariate time series $f[(v_i, t)]$, $i = 1, \cdots, 20$, confirmed cases of COVID-19 per 100,000 people every week (referred to as confirmed cases) for each of the G20:

$$f[(v_i, t)] = 100000 * I_i(t)/C_i, \quad t = 1, \cdots, T, \tag{31}$$

where $I_i(t)$ is the number of confirmed cases every week in country $v_i$ at time $t$ and $C_i$ is the population of country $v_i$. Let $T = 55$. The time evolution of $f[(v_i, t)]$, $i = 1, \cdots, 20$, are plotted in Fig. 10 for G20.

For the multivariate time series of G20 confirmed cases, there is no graph structure at all. Since the correlation of COVID-19 data between each country is different and changes over time. It may be affected by complex factors such as trade or traffic flows among these countries. Therefore, we first construct a sequence of identical complete graphs $\{G_t, t = 1, \cdots, T\}$ without self-loops. At this time, there is no weight on the graph $G_t$. Then we obtain the transformed graph $\mathcal{G}_T$ using the temporal-attention product. To better capture the spatio-temporal connection of nodes, i.e., the weight in the transformed graph, the graph attention neural network (GAT) [50] is used to learn the time-dependent attention weight of each pair of nodes. GAT is a powerful deep learning method [50,67–70], which has state-of-art performance on node classification, edge classification, and link predictions. The learned attention (weight) matrix is bidirectional on the transformed graph. It is worth noting that, according to the spread of the pandemic, time flows in one direction, from time $t$ to time $t + 1$; only the weights of temporal-attention edges from $G_t$ to $G_{t+1}$ need to be learned, while the weights of temporal-attention edges from $G_{t+1}$ to $G_t$ are 0.

[2] https://covid19.who.int.

### 6.2.1. *Learning spatio-temporal transition matrix $\mathcal{P}_T$ in the transformed graph*

We introduce how to use GAT to learn the weights of transformed graphs based on the topological structure and COVID-19 data feature similarity. We take the GAT coefficient obtained from the last layer of the graph neural network as the weight of the transformed graph. This gives the spatio-temporal transition matrix $\mathcal{P}_T = (p_{(\mathbf{v}_i,\mathbf{v}_j)})_{N_T^\# \times N_T^\#}$ as the weighted matrix in our transformed graph $\mathcal{G}_T = (\mathcal{V}_T, \mathcal{E}_T)$.

**Review the architecture of GAT.**

We review the architecture of GAT. Let $X_{\mathbf{v}_i} \in \mathbb{R}^d$ be a $d$-dimensional feature of node $\mathbf{v}_i$ in the transformed graph $\mathcal{G}_T$. Let $H_{\mathbf{v}_i}^{(1)} = X_{\mathbf{v}_i}$ be the initial feature at node $\mathbf{v}_i$. To increase the accuracy, one can add several layers to the graph neural network. At the $l$-th layer, $l \geq 1$, $H_{\mathbf{v}_i}^{(l)}$ denotes the input feature with dimension $d_l$. $H_{\mathbf{v}_i}^{(l+1)}$ is the output feature with dimension $d_{l+1}$. The relative importance of neighboring features to each other can be described by attention coefficients, which are calculated using an attentional function $\alpha_{(\mathbf{v}_i,\mathbf{v}_j)}^{(l)} : \mathbb{R}^{d_l} \times \mathbb{R}^{d_l} \to \mathbb{R}$:

$$\alpha_{(\mathbf{v}_i,\mathbf{v}_j)}^{(l)} = \frac{\exp\Big(\mathrm{LeakyReLU}(\hat{a} \cdot (W^{(l)} H_{\mathbf{v}_i}^{(l)} \| W^{(l)} H_{\mathbf{v}_j}^{(l)}))\Big)}{\sum_{\mathbf{v}_k \in \mathcal{N}_{\mathbf{v}_i}^{1,T}} \exp\Big(\mathrm{LeakyReLU}(\hat{a} \cdot (W^{(l)} H_{\mathbf{v}_i}^{(l)} \| W^{(l)} H_{\mathbf{v}_k}^{(l)}))\Big)}, \tag{32}$$

where $\|$ is the concatenation operation, and $W^{(l)} \in \mathbb{R}^{d_{l+1} \times d_l}$ is a learned embedding matrix. $\hat{a} \in \mathbb{R}^{2d_{l+1}}$ is a single-layer feed-forward neural network. LeakyReLU[3] is the activation function. Moreover, we assign $\alpha_{(\mathbf{v}_i,\mathbf{v}_i)}^{(l)} = 0$ to avoid self-loop.

$$H_{\mathbf{v}_i}^{(l+1)} = \mathrm{ELU}\Bigg(\sum_{\mathbf{v}_j \in \mathcal{N}_{\mathbf{v}_i}^{1,T}} \alpha_{(\mathbf{v}_i,\mathbf{v}_j)}^{(l)} W^{(l)} H_{\mathbf{v}_j}^{(l)}\Bigg), \tag{33}$$

where ELU is an exponential linear unit.[4] To improve the stability of the learning process, we use a multi-head attention mechanism, that is, run the above process several times in parallel. The output features of nodes' neighbors are either concatenated or averaged to form their final output features.

Let $S = S_+ \cup S_-$ be the sample set of edges for the transformed graph, where $S_+$ is a positive sample set composed of node pairs with connected edges, and $S_-$ is a negative sample set composed of node pairs without connected edges. The whole GAT framework is trained by minimizing the following cost function:

$$\mathcal{L} = -(1/|S|) \sum_{(\mathbf{v}_i,\mathbf{v}_j) \in S} \left(y_{i,j} \log r_{ij} + (1 - y_{i,j}) \log(1 - r_{ij})\right), \tag{34}$$

where $|\cdot|$ represents the cardinality of the set. $y_{i,j}$ is the label information for the edge between $\mathbf{v}_i$ and $\mathbf{v}_j$, with 0 for non-existence and 1 for existence.

$$r_{i,j} = \mathrm{Sigmoid}\Big(\theta \cdot (H_{\mathbf{v}_i}^{(l+1)} \odot H_{\mathbf{v}_j}^{(l+1)})\Big), \tag{35}$$

represents the probability of whether there is an edge between node $\mathbf{v}_i$ and node $\mathbf{v}_j$. Sigmoid[5] is an activation function. $\odot$ denotes Hadamard product. $\theta$ is the learned parameter vector with the dimension as same as $H_{\mathbf{v}_i}^{(l+1)}$. This setting makes the feature outputs after GAT more relevant if there is an edge between two nodes.

[3] $\mathrm{LeakyReLU}(x) = 0.15x(1 - \mathrm{sgn}(x))/2 + x(1 + \mathrm{sgn}(x))/2$.

[4] $\mathrm{ELU}(x) = (\exp(x) - 1)(1 - \mathrm{sgn}(x))/2 + x(1 + \mathrm{sgn}(x))/2$.

[5] $\mathrm{Sigmoid}(x) = 1/(1 + \exp(-x))$.

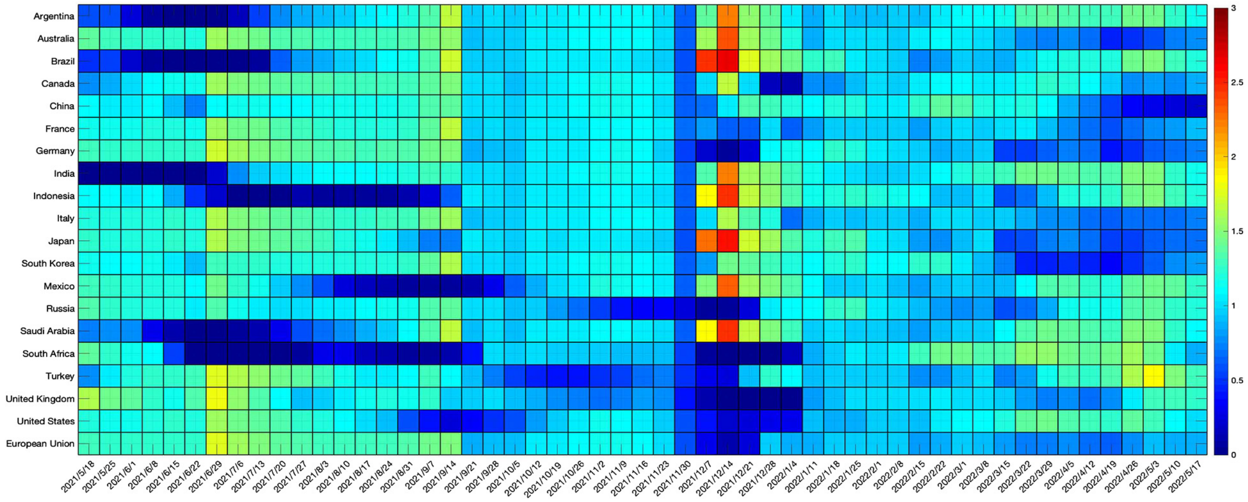


**Fig. 11.** The time evolution of the pandemic centrality score $c_i(t)$ in G20. The color represents the value of the centrality score.

**Neural network training settings.**

In our setup, the graph signal $f[(v,t)]$ only takes one value at each node. This makes the learning of attention weight rather difficult. To overcome this difficulty, we design a new feature function on the transformed graph, which carries short-term temporal information. More precisely, for each node $(v,t) \in V_t$, we define a 4-dimension $X_{(v,t)} = (f[(v,t-3)], \cdots, f[(v,t)])$ as the new temporal feature at node $(v,t)$ in the transformed graph $\mathcal{G}_T$. Therefore, in addition to the period we analyzed, we also use data for three weeks from April 20, 2021 to May 4, 2021.

We take all connected node pairs in the transformed graph (which is rather sparse) as positive samples, and downsample the same unconnected node pairs as negative samples. The ratio of the training set, validation set, and test set is 6:2:2. The number of positive samples and negative samples in the test set is the same. We use two layers of graph attention layers, the first layer has 7 attention heads, and the hidden layer has 84 units and the output is in the form of a concatenation of 7 attention heads. The second layer has a single attention head, and the hidden layer has 52 units. The transition probability $p_{(\mathbf{v}_i,\mathbf{v}_j)}$ is defined as the GAT coefficients of the last GAT layer, i.e., (32) with $l = 2$. During the training process, the learning rate is $1e-5$, the dropout rate is 0.1, and the optimizer is Adam. The early stopping strategy is applied to the validation set to avoid overfitting, with the patience sets to 50 epochs. The edge classification accuracy for the test set is 0.9534.

### 6.2.2. *Capturing centrality of each country over time*

The attention probability $p_{(\mathbf{v}_i,\mathbf{v}_j)}$ on $\mathcal{G}_T$ has the advantage to capture the importance of the characteristics of $\mathbf{v}_j$ to $\mathbf{v}_i$ among all its neighbors on $\mathcal{G}_T$. This enables us to identify the centrality of the $i$-th country at time $t$ using the following formula:

$$c_i(t) = \sum_{\mathbf{w} \in \mathcal{N}^{1,T}_{(v_i,t)}} p_{(w,v_i,t)}, \quad i = 1, \cdots, 20. \tag{36}$$

It is the sum of the in-degree weights for each country and changes over time.

We demonstrate the time evolution of the centrality score $c_i(t)$ in G20 in Fig. 11. Since the time slices of the endpoints lack neighbors on one side, we omit the endpoint time slice in our subsequent visualization analysis. The centrality score changes over time. Take December 14, 2021 as an example, the number of confirmed cases increased sharply during this time, making the centrality score change more obvious. The countries with top five centrality scores were Brazil, Japan, Indonesia, Sandi Arabia, and Australia. It can

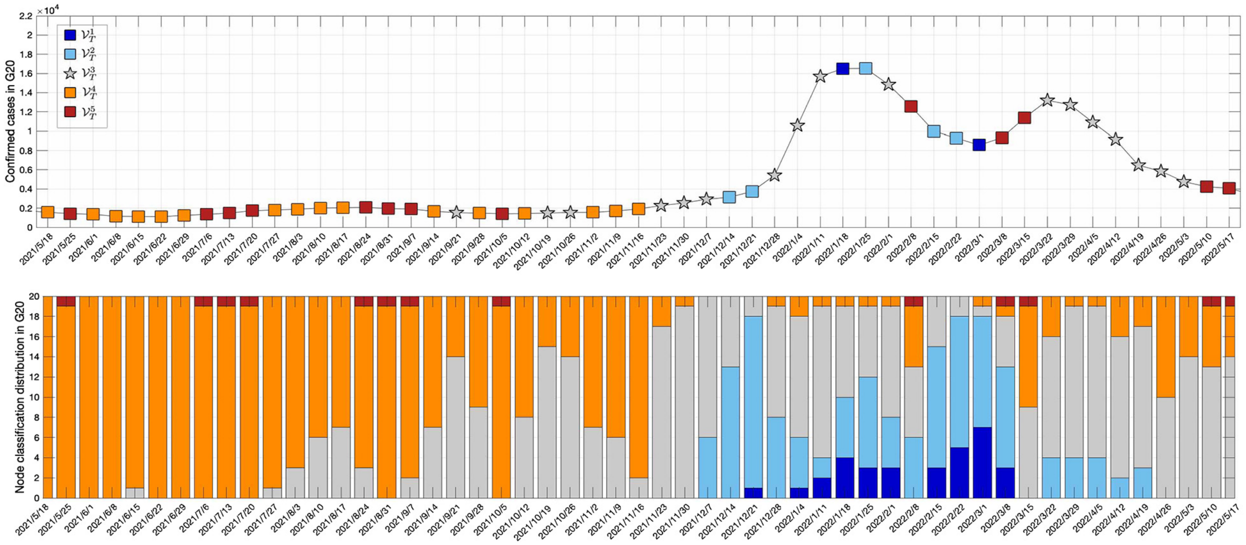


**Fig. 12.** Overall visual analysis using node classification and graph classification. The top figure depicts graph classification for each graph time slice by $\sigma_t$ defined in (29). The bottom figure depicts the node classification distribution $\rho_t^i$ given by (28).

be seen from the inset in Fig. 10 that the pandemic situation in these countries began to show an upward trend in the following month. Due to the delayed nature of pandemic transmission, the pandemic in some countries will begin to increase subsequently. The pandemic has shown a downward trend in countries with low centralities, such as Germany and Russia. For the United Kingdom and the United States, the pandemic was on the rise at the beginning of December 14 due to the previous high centrality effect. Still, after a while, their pandemic began to decline.

### *6.2.3. Overall visual analysis*

We conduct an overall analysis of the spatio-temporal changes of the pandemic in G20 by the graph classification method after performing SGWT.

As shown in Fig. 12, most of the time during the pandemic, the change in severity in each country was uneven. The seriousness in some countries was significantly higher or lower than in others. This makes the detected signal change mainly belong to mid-high frequency ($\mathcal{V}_T^4$) or high frequency ($\mathcal{V}_T^5$). This is also consistent with Fig. 10. The time slice from December 14, 2021 to December 21, 2021 is classified as $\mathcal{V}_T^2$. Due to the spread of the Omicron variant, the pandemic in most countries was on the rise, and the spatial and temporal changes were low frequency. On January 18, 2022, the time slice is classified as $\mathcal{V}_T^1$. During this period, only the French pandemic was high-frequency, and most countries had similar temporal and spatial changes, so the proportion of countries classified as low-frequency is large, and the time slice was classified as low-frequency. It can be seen that the rapid spread of the Omicron variant makes the pandemic between countries more similar. Then, possibly due to different policies adopted by countries, the changes in severity began to differ. Therefore, a subsequent time slice is classified as $\mathcal{V}_T^5$.

### *6.2.4. Ranking of G20 using the anomaly scores*

We first evaluate the pandemic severity in G20 using the refined classification method – a-scores $\chi(v,t)$ with the hyperparameters $\zeta_1 = 1.5$ and $\zeta_2 = 0.5$. We define the average anomaly score:

$$\bar{\chi}(v) := (1/T)\sum_{t=1}^{T}\chi[(v,t)]. \tag{37}$$

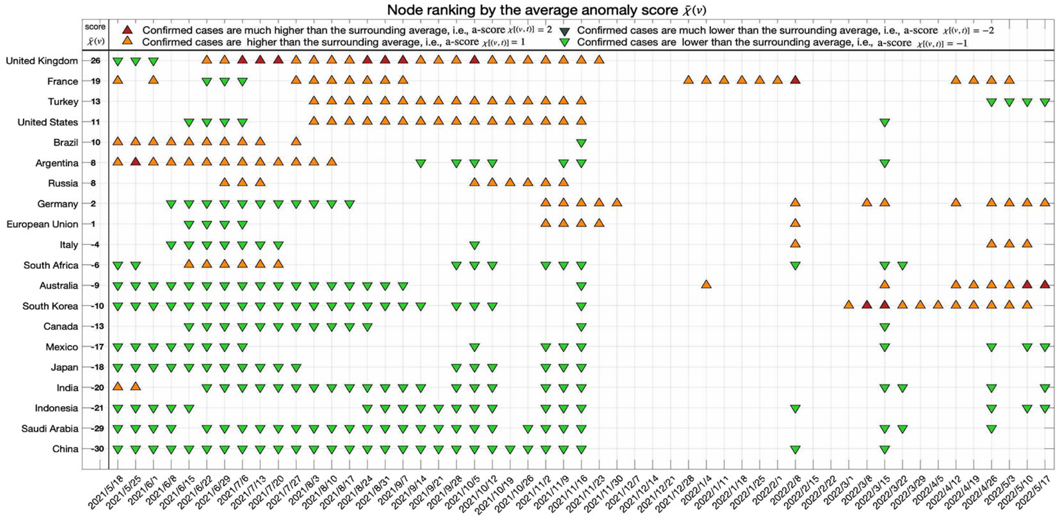


**Fig. 13.** Ranking of pandemic severity using the average a-score in G20. The first column is the average anomaly score $\bar{\chi}(v)$ defined in (37). $\chi[(v,t)]$ is the a-score defined in (27).

Then based on the average a-scores $\bar{\chi}(v)$, we give overall ranking of these countries from May 18, 2021 to May 17, 2022 as follows: the United Kingdom, France, Turkey, the United States, Brazil, Argentina, Russia, Germany, European Union, Italy, South Africa, Australia, South Korea, Canada, Mexico, Japan, India, Indonesia, Saudi Arabia and China, see Fig. 13. The ranking provides a clear understanding of the pandemic virus spread information among these countries. This is helpful to analyze the pandemic spread patterns and find countries with good pandemic protection policies (such as Indonesia), and provide a good reference for further pandemic prevention and control. Additionally, we can examine each country's pandemic intensity throughout a specific period from Fig. 13. For example, the countries with severe pandemics from January 4, 2022 to May 17, 2022 are South Korea, Australia, Germany, France, and Italy. On the other hand, the countries with fewer pandemics are China, Saudi Arabia and Indonesia.

## 7. Conclusion

In this paper, we propose a new method for modeling spatio-temporal dynamic graphs, by introducing the temporal-attention product, which better reflects the evolution of the signal concerning the surrounding environment over time. Especially, for dynamic networks with topological changes, this connection method fully considers the spatial and temporal topological connections. The newly added temporal edges will not affect the previous spatial graph structures. This method is robust, scalable, and can be defined inductively. The transformed graph provides a solid mathematical foundation to model the time-dependent graph signal processes as martingales. Moreover, the transformed graph is highly sparse. The number of edges of $\mathcal{G}_T$ is at most twice the total edges of the sequence of the dynamic graph network $\cup_{t=1}^{T} G_t$. The large weighted adjacency matrix obtained by the temporal-attention product is a block tridiagonal matrix, which appears in many applications and has been extensively studied. Thus, it is very convenient to study its spectral properties.

Based on the transformed graph, we can extend GFT and SGWT to study dynamic graph signals, as well as multivariate time series. Compared with the previous dynamic graph signal processing methods, the changes in topology and nodes of the transformed graph can be incorporated inductively. It is conducive to the analysis of data flow. We apply two real datasets for spectral graph wavelet visualization analysis. For the COVID-19 confirmed data of the G20 without a graph structure, we use GAT and COVID-19 confirmed data to learn the weighted matrix of the transformed graph composed of complete graphs. We design different visualization diagrams and methods for selecting key nodes, and provide valuable information for

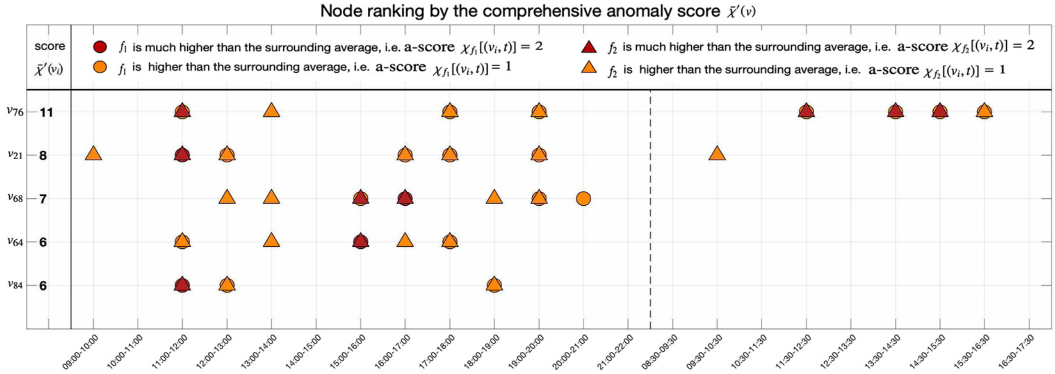


**Fig. A.14.** Node ranking by temporal-attention product. The circle corresponds to the signal $f_1$, and the triangle corresponds to the signal $f_2$. The dotted line marks the start of the second-day meeting. The first column is the score $\bar{\chi}'(v)$ defined in (30). $\chi_{f_i}[(v,t)]$ is the a-score defined in (27).

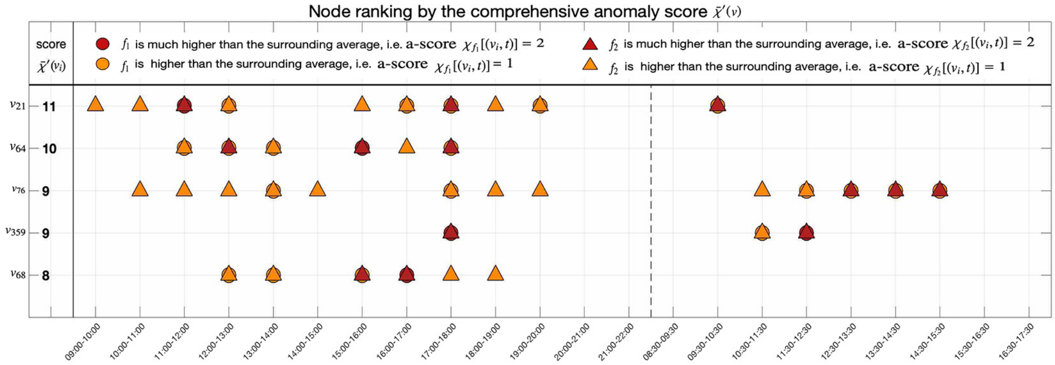


**Fig. A.15.** Node ranking by Cartesian product. The circle corresponds to the signal $f_1$, and the triangle corresponds to the signal $f_2$. The dotted line marks the start of the second-day meeting. The first column is the score $\bar{\chi}'(v)$ defined in (30). $\chi_{f_i}[(v,t)]$ is the a-score defined in (27).

signal analysis. For example, it can identify the most popular participants in different social circles using the SFHH conference dataset. Another example, from the COVID-19 data sets of G20, we found countries with severe pandemic spread patterns, and identify countries with good pandemic prevention. These can be used as references for further analysis of pandemics and the formulation of pandemic prevention policies. It can be seen that SGWT on the transformed graph can successfully analyze the spatio-temporal dynamic properties of events and capture the key information of interest.

## Data availability

This article uses two datasets: the 2009 SFHH conference in Nice and COVID-19 confirmed cases in G20. We have given the source link in the article.

## Appendix A. Comparison of the temporal-attention product, Cartesian products, and strong product

To demonstrate that SGWT on the dynamic graph connected by temporal-attention product can better capture the spatio-temporal dynamic properties of the signal, we also employ SGWT on the dynamic graph connected by Cartesian product and strong product for case study 2 in section 6.1. Participants are ranked for popularity using the same method as subsection 6.1.3. Fig. A.14, Fig. A.15 and Fig. A.16 show the node ranking results on the dynamic graph connected by the temporal-attention product, Cartesian product and

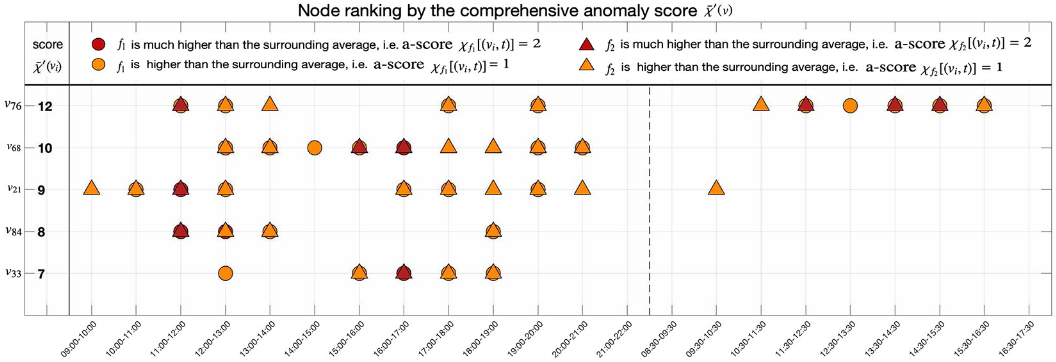


**Fig. A.16.** Node ranking by strong product. The circle corresponds to the signal $f_1$, and the triangle corresponds to the signal $f_2$. The dotted line marks the start of the second-day meeting. The first column is the score $\bar{\chi}'(v)$ defined in (30). $\chi_{f_i}[(v,t)]$ is the a-score defined in (27).

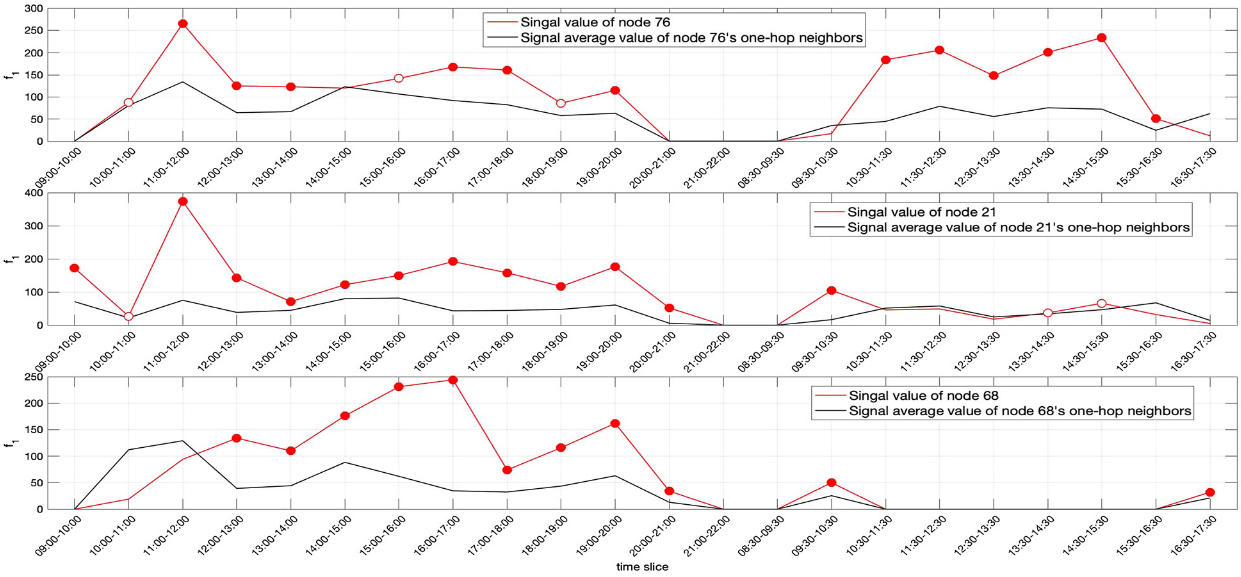


**Fig. A.17.** The node signal value $f_1$ and the average signal value of its neighbors. A hollow circle represents a node whose signal value exceeds the average value of its neighbors, and a solid circle represents a node whose signal value exceeds 1.5 times the average signal value of its neighbors.

strong product, respectively. We choose three nodes $v_{76}, v_{21}$ and $v_{68}$ to analyze which of the three rankings is more reasonable.

Because the idea behind our popularity ranking is to assign an a-score by comparing the signal value of the node with the average value of its one-hop neighbor nodes after the SGWT identifies that the node belongs to high frequency or mid-high frequency. The comprehensive a-score of a node is calculated by multiplying the a-scores of the node's two signals and taking the average value over time. If a node has a higher comprehensive a-score, it means that not only the number of people talking to this node is more than the average number of interactions between its neighbors and other people, but also the conversation time is longer than the average value of interactions between its neighbors and other people. This indicates that more people tend to spend more time interacting with this node, thus showing that the node is more popular in the venue.

According to the idea of popularity ranking, we can test the most popular nodes by two criteria:

(i). The nodes with signal value $f_i[(v,t)] > \bar{f}_i[(v,t)]$, $i = 1, 2$,

(ii). The nodes with signal value $f_i[(v,t)] > 1.5 \cdot \bar{f}_i[(v,t)]$, $i = 1, 2$,

where $\bar{f}_i[(v,t)]$, defined in (26), is the average signal value of one-hop neighbors of the node $(v,t)$ on $G_t$.

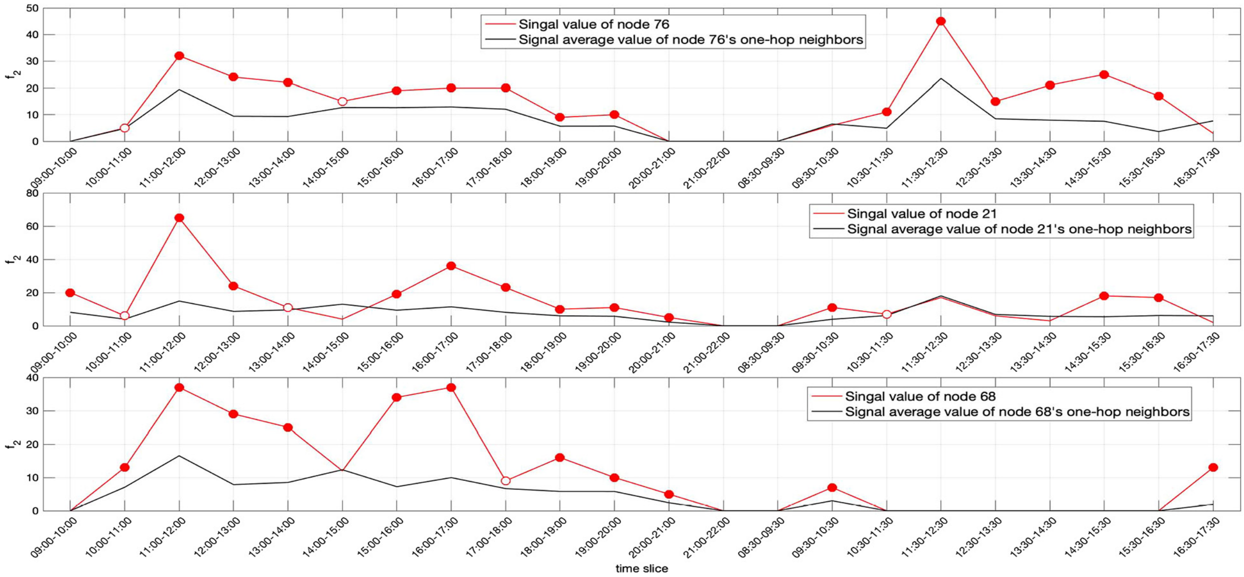


**Fig. A.18.** The node signal value $f_2$ and the average signal value of its neighbors. A hollow circle represents a node whose signal value exceeds the average value of its neighbors, and a solid circle represents a node whose signal value exceeds 1.5 times the average signal value of its neighbors.

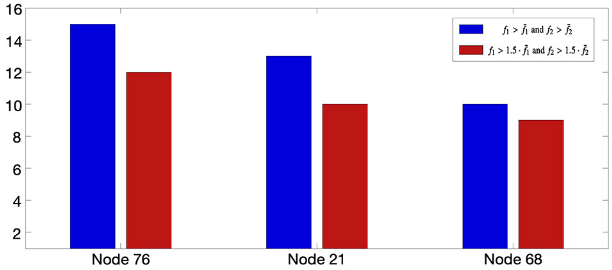


**Fig. A.19.** The number of time slices where both signals $f_1 > \bar{f}_1$ and $f_2 > \bar{f}_2$ is shown in blue bars, while the number of time slices where both signals $f_1 > 1.5 \cdot \bar{f}_1$ and $f_2 > 1.5 \cdot \bar{f}_2$ is shown in red bars.

Fig. A.17 and Fig. A.18 depict the signal $f_i$ of nodes $v_{76}, v_{21}$ and $v_{68}$, and the corresponding $\bar{f}_i$, respectively. Fig. A.19 shows the number of time slices that $f_1 > \bar{f}_1$ and $f_2 > \bar{f}_2$ (blue bars), and the number of time slices that $f_1 > 1.5 \cdot \bar{f}_1$ and $f_2 > 1.5 \cdot \bar{f}_2$ (red bars). It can be seen that $v_{76}$ should be ranked ahead of $v_{21}$ in terms of popularity, and $v_{21}$ should be ranked ahead of node $v_{68}$. Therefore the ranking in Fig. 8 is more reasonable.

The comparison of SGWT of different connection methods in this section is rough, due to the signal criteria being self-defined. It would be interesting for future work to conduct a deeper analysis of this comparison. The weights of the majority of temporal edges in the strong product graph are defined as the same as the temporal-attention product method, i.e., $\mathbf{w}_{((v,t),(w,t+1))} := w_{(v,w,t)}$, $\mathbf{w}_{((v,t),(w,t))} := w_{(v,w,t)}$. But for the edge between the same base node on the adjacent time slice, similar to [23], the weight is assigned as 1 for the Cartesian product and strong product. Using different weights may yield different results. However, since there is no self-loop in the original time slice $G_t$, what weight is reasonable is a question that needs further discussion for the Cartesian product and strong product. It can be seen that the ability of the temporal-attention product to assist SGWT in finding important and interesting information in the applications we exemplify. Therefore, the temporal-attention product is reasonable compared to the other methods.

## References

[1] A. Ortega, P. Frossard, J. Kovačević, J.M. Moura, P. Vandergheynst, Graph signal processing: overview, challenges, and applications, Proc. IEEE 106 (5) (2018) 808–828, https://doi.org/10.1109/JPROC.2018.2820126.

[2] S. Itani, D. Thanou, A graph signal processing framework for the classification of temporal brain data, in: 2020 28th European Signal Processing Conference, IEEE, 2021, pp. 1180–1184.

[3] G. Taubin, A signal processing approach to fair surface design, in: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, 1995, pp. 351–358.

[4] A. Agaskar, Y.M. Lu, A spectral graph uncertainty principle, IEEE Trans. Inf. Theory 59 (7) (2013) 4338–4356, https://doi.org/10.1109/TIT.2013.2252233.

[5] A. Sandryhaila, J.M. Moura, Discrete signal processing on graphs: frequency analysis, IEEE Trans. Signal Process. 62 (12) (2014) 3042–3054, https://doi.org/10.1109/TSP.2014.2321121.

[6] D.I. Shuman, B. Ricaud, P. Vandergheynst, Vertex-frequency analysis on graphs, Appl. Comput. Harmon. Anal. 40 (2) (2016) 260–291, https://doi.org/10.1016/j.acha.2015.02.005.

[7] R. Balan, I. Daubechies, V. Vaishampayan, The analysis and design of windowed Fourier frame based multiple description source coding schemes, IEEE Trans. Inf. Theory 46 (7) (2000) 2491–2536, https://doi.org/10.1109/18.887860.

[8] D.K. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory, Appl. Comput. Harmon. Anal. 30 (2) (2011) 129–150, https://doi.org/10.1016/j.acha.2010.04.005.

[9] D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, 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. 30 (3) (2013) 83–98, https://doi.org/10.1109/MSP.2012.2235192.

[10] D.I. Shuman, C. Wiesmeyr, N. Holighaus, P. Vandergheynst, Spectrum-adapted tight graph wavelet and vertex-frequency frames, IEEE Trans. Signal Process. 63 (16) (2015) 4223–4235, https://doi.org/10.1109/TSP.2015.2424203.

[11] D.M. Mohan, M.T. Asif, N. Mitrovic, J. Dauwels, P. Jaillet, Wavelets on graphs with application to transportation networks, in: 17th International IEEE Conference on Intelligent Transportation Systems, IEEE, 2014, pp. 1707–1712.

[12] N. Tremblay, P. Borgnat, Graph wavelets for multiscale community mining, IEEE Trans. Signal Process. 62 (20) (2014) 5227–5239, https://doi.org/10.1109/TSP.2014.2345355.

[13] P. Valdivia, F. Dias, F. Petronetto, C.T. Silva, L.G. Nonato, Wavelet-based visualization of time-varying data on graphs, in: 2015 IEEE Conference on Visual Analytics Science and Technology, IEEE, 2015, pp. 1–8.

[14] B. Dong, Q. Jiang, C. Liu, Z. Shen, Multiscale representation of surfaces by tight wavelet frames with applications to denoising, Appl. Comput. Harmon. Anal. 41 (2) (2016) 561–589, https://doi.org/10.1016/j.acha.2015.03.005.

[15] G.W. Yu, X. Zhuang, Tight framelets and fast framelet transforms on manifolds, Appl. Comput. Harmon. Anal. 48 (1) (2016) 64–95, https://doi.org/10.1016/j.acha.2018.02.001.

[16] F. Harary, G. Gupta, Dynamic graph models, Math. Comput. Model. 25 (7) (1997) 79–87, https://doi.org/10.1016/S0895-7177(97)00050-2.

[17] S. Moreno, S. Kirshner, J. Neville, S. Vishwanathan, Tied Kronecker product graph models to capture variance in network populations, in: 2010 48th Annual Allerton Conference on Communication, Control, and Computing, IEEE, 2010, pp. 1137–1144.

[18] S.I. Moreno, J. Neville, S. Kirshner, Learning mixed Kronecker product graph models with simulated method of moments, in: Acm Sigkdd International Conference on Knowledge Discovery and Data Mining, 2013, pp. 1052–1060.

[19] A. Sandryhaila, J.M. Moura, Big data analysis with signal processing on graphs: representation and processing of massive data sets with irregular structure, IEEE Signal Process. Mag. 31 (5) (2014) 80–90, https://doi.org/10.1109/MSP.2014.2329213.

[20] A.D. Col, P. Valdivia, F. Petronetto, F. Dias, C.T. Silva, L.G. Nonato, Wavelet-based visual analysis for data exploration, Comput. Sci. Eng. 19 (5) (2017) 85–91, https://doi.org/10.1109/MCSE.2017.3421553.

[21] M. Villafañe-Delgado, S. Aviyente, Dynamic graph Fourier transform on temporal functional connectivity networks, in: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2017, pp. 949–953.

[22] F. Grassi, N. Perraudin, B. Ricaud, Tracking time-vertex propagation using dynamic graph wavelets, in: 2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP), IEEE, 2016, pp. 351–355.

[23] A. Dal Col, P. Valdivia, F. Petronetto, F. Dias, C.T. Silva, L.G. Nonato, Wavelet-based visual analysis of dynamic networks, IEEE Trans. Vis. Comput. Graph. 24 (8) (2017) 2456–2469, https://doi.org/10.1109/MCSE.2017.3421553.

[24] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, Y. Bengio, End-to-end attention-based large vocabulary speech recognition, in: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2016, pp. 4945–4949.

[25] A. Asif, J.M. Moura, Data assimilation in large time-varying multidimensional fields, IEEE Trans. Image Process. 8 (11) (1999) 1593–1607, https://doi.org/10.1109/83.799887.

[26] E. Galligani, V. Ruggiero, A polynomial preconditioner for block tridiagonal matrices, Parallel Algorithms Appl. 3 (3–4) (1994) 227–237, https://doi.org/10.1080/10637199408962539.

[27] I. Braeutigam, D.M. Polyakov, Asymptotics of eigenvalues of infinite block matrices, Ufa Math. J. 11 (3) (2019) 11–28, https://doi.org/10.13108/2019-11-3-11.

[28] G. Casati, I. Guarneri, F.M. Izrailev, L. Molinari, K. Życzkowski, Periodic band random matrices, curvature, and conductance in disordered media, Phys. Rev. Lett. 72 (17) (1994) 2697, https://doi.org/10.1103/PhysRevLett.72.2697.

[29] B. Kramer, A. MacKinnon, Localization: theory and experiment, Rep. Prog. Phys. 56 (12) (1993) 1469, https://doi.org/10.1088/0034-4885/56/12/001.

[30] D.E. Petersen, H.H.B. Sørensen, P.C. Hansen, S. Skelboe, K. Stokbro, Block tridiagonal matrix inversion and fast transmission calculations, J. Comput. Phys. 227 (6) (2008) 3174–3190, https://doi.org/10.1016/j.jcp.2007.11.035.

[31] H. Dette, B. Reuther, W. Studden, M. Zygmunt, Matrix measures and random walks with a block tridiagonal transition matrix, SIAM J. Matrix Anal. Appl. 29 (1) (2007) 117–142, https://doi.org/10.1137/050638230.

[32] F.A. Grünbaum, The Karlin–Mcgregor formula for a variant of a discrete version of Walsh's spider, J. Phys. A, Math. Theor. 42 (45) (2009) 454010, https://doi.org/10.1088/1751-8113/42/45/454010.
[33] S. Iida, H. Weidenmüller, J. Zuk, Statistical scattering theory, the supersymmetry method and universal conductance fluctuations, Ann. Phys. 200 (2) (1990) 219–270, https://doi.org/10.1016/0003-4916(90)90275-S.
[34] J.D. Anderson, J. Wendt, Computational Fluid Dynamics, vol. 206, Springer, 1995.
[35] A. Kavcic, J.M. Moura, Matrices with banded inverses: inversion algorithms and factorization of Gauss-Markov processes, IEEE Trans. Inf. Theory 46 (4) (2000) 1495–1509, https://doi.org/10.1109/18.954748.
[36] J.M. Moura, N. Balram, Recursive structure of noncausal Gauss-Markov random fields, IEEE Trans. Inf. Theory 38 (2) (1992) 334–354, https://doi.org/10.1109/18.119691.
[37] W.N. Gansterer, R.C. Ward, R.P. Muller, An extension of the divide-and-conquer method for a class of symmetric block-tridiagonal eigenproblems, ACM Trans. Math. Softw. 28 (1) (2002) 45–58, https://doi.org/10.1145/513001.513004.
[38] Y. Bai, W.N. Gansterer, R.C. Ward, Block tridiagonalization of 'effectively' sparse symmetric matrices, ACM Trans. Math. Softw. 30 (3) (2004) 326–352, https://doi.org/10.1145/1024074.1024078.
[39] W.N. Gansterer, R.C. Ward, R.P. Muller, W.A. Goddard, Computing approximate eigenpairs of symmetric block tridiagonal matrices, SIAM J. Sci. Comput. 25 (1) (2003) 65–85, https://doi.org/10.1137/s1064827501399432.
[40] W.N. Gansterer, Computing orthogonal decompositions of block tridiagonal or banded matrices, in: International Conference on Computational Science, Springer, 2005, pp. 25–32.
[41] W.N. Gansterer, J. Zottl, Parallelization of divide-and-conquer eigenvector accumulation, in: European Conference on Parallel Processing, Springer, 2005, pp. 847–856.
[42] Y. Bai, R.C. Ward, A parallel symmetric block-tridiagonal divide-and-conquer algorithm, ACM Trans. Math. Softw. 33 (4) (2007) 25-es, https://doi.org/10.1145/1268776.1268780.
[43] G. König, M. Moldaschl, W.N. Gansterer, Computing eigenvectors of block tridiagonal matrices based on twisted block factorizations, J. Comput. Appl. Math. 236 (15) (2012) 3696–3703, https://doi.org/10.1016/j.cam.2011.07.010.
[44] W.N. Gansterer, G. König, On twisted factorizations of block tridiagonal matrices, Proc. Comput. Sci. 1 (1) (2010) 279–287, https://doi.org/10.1016/j.procs.2010.04.031.
[45] R. Askey, Orthogonal Polynomials and Special Functions, SIAM, 1975.
[46] W. Gautschi, Orthogonal Polynomials: Computation and Approximation, OUP, Oxford, 2004.
[47] A. Sandryhaila, J. Kovacevic, M. Puschel, Algebraic signal processing theory: 1-d nearest neighbor models, IEEE Trans. Signal Process. 60 (5) (2012) 2247–2259, https://doi.org/10.1109/TSP.2012.2186133.
[48] A.J. Duran, P. Lopez-Rodriguez, Orthogonal matrix polynomials: zeros and Blumenthal's theorem, J. Approx. Theory 84 (1) (1996) 96–118, https://doi.org/10.1006/jath.1996.0007.
[49] A. Sandryhaila, J.M. Moura, Eigendecomposition of block tridiagonal matrices, arXiv preprint, arXiv:1306.0217, 2013.
[50] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, in: International Conference on Learning Representations, 2018.
[51] W.-C. Yueh, Eigenvalues of several tridiagonal matrices, Appl. Math. E-Notes [electronic only] 5 (2005) 66–74.
[52] A.J. Laub, Matrix Analysis for Scientists and Engineers, vol. 91, SIAM, 2005.
[53] W.N. Gansterer, Y. Bai, R.M. Day, R.C. Ward, Framework for approximating eigenpairs in electronic structure computations, Comput. Sci. Eng. 6 (5) (2004) 50–59, https://doi.org/10.1109/MCSE.2004.25.
[54] R. Geng, Y. Gao, H. Zhang, J. Zu, Analysis of the spatio-temporal dynamics of Covid-19 in Massachusetts via spectral graph wavelet theory, IEEE Trans. Signal Inf. Process. Netw. (2022), https://doi.org/10.1109/TSIPN.2022.3193252.
[55] A.C. Müller, S. Guido, Introduction to Machine Learning with Python: a Guide for Data Scientists, O'Reilly Media, Inc., 2016.
[56] H. Yao, C. Jiang, Y. Qian, Developing Networks Using Artificial Intelligence, Springer, 2019.
[57] J.L. Sharpnack, A. Krishnamurthy, A. Singh, Near-optimal anomaly detection in graphs using Lovasz extended scan statistic, Adv. Neural Inf. Process. Syst. 26 (2013), https://doi.org/10.1016/j.asoc.2022.108489.
[58] K. Sricharan, K. Das, Localizing anomalous changes in time-evolving graphs, in: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, 2014, pp. 1347–1358.
[59] M. Génois, A. Barrat, Can co-location be used as a proxy for face-to-face contacts?, EPJ Data Sci. 7 (1) (2018) 11, https://doi.org/10.1140/EPJDS/S13688-018-0140-1.
[60] R.M. Anderson, H. Heesterbeek, D. Klinkenberg, T.D. Hollingsworth, How will country-based mitigation measures influence the course of the Covid-19 epidemic?, Lancet 395 (10228) (2020) 931–934, https://doi.org/10.1016/SO140-6736(20)30567-5.
[61] R.A. Middelburg, F.R. Rosendaal, Covid-19: how to make between-country comparisons, Int. J. Infect. Dis. 96 (2020) 477–481, https://doi.org/10.1016/j.ijid.2020.05.066.
[62] B. Balmford, J.D. Annan, J.C. Hargreaves, M. Altoè, I.J. Bateman, Cross-country comparisons of Covid-19: policy, politics and the price of life, Environ. Resour. Econ. 76 (4) (2020) 525–551, https://doi.org/10.1007/s10640-020-00466-5.
[63] M. Rafiq, S.H. Batool, A.F. Ali, M. Ullah, University libraries response to Covid-19 pandemic: a developing country perspective, J. Acad. Librariansh. 47 (1) (2021) 102280, https://doi.org/10.1016/j.acalib.2020.102280.
[64] P. Tarkar, Impact of Covid-19 pandemic on education system, Int. J. Adv. Sci. Technol. 29 (9) (2020) 3812–3814, https://doi.org/10.36713/epra6363.
[65] D.H.B. Phan, P.K. Narayan, Country responses and the reaction of the stock market to Covid-19—a preliminary exposition, Emerg. Mark. Finance Trade 56 (10) (2020) 2138–2150, https://doi.org/10.1080/1540496X.2020.1784719.
[66] I. Djekic, A. Nikolić, M. Uzunović, A. Marijke, A. Liu, J. Han, M. Brnčić, N. Knežević, P. Papademas, K. Lemoniati, et al., Covid-19 pandemic effects on food safety-multi-country survey study, Food Control 122 (2021) 107800, https://doi.org/10.1016/j.foodcont.2020.107800.
[67] W. Gu, F. Gao, X. Lou, J. Zhang, Link prediction via graph attention network, arXiv preprint, arXiv:1910.04807, 2019.
[68] C. Tang, J. Sun, Y. Sun, M. Peng, N. Gan, A general traffic flow prediction approach based on spatial-temporal graph attention, IEEE Access 8 (2020) 153731–153741, https://doi.org/10.1109/ACCESS.2020.3018452.

[69] H. Zhou, D. Ren, H. Xia, M. Fan, X. Yang, H. Huang, Ast-gnn: an attention-based spatio-temporal graph neural network for interaction-aware pedestrian trajectory prediction, Neurocomputing 445 (2021) 298–308, https://doi.org/10.1016/j.neucom.2021.03.024.

[70] Z. Zhang, J. Huang, Q. Tan, Sr-hgat: symmetric relations based heterogeneous graph attention network, IEEE Access 8 (2020) 165631–165645, https://doi.org/10.1109/ACCESS.2020.3022664.