Interpreting Temporal Graph Neural Networks with Koopman Theory

Anonymous authors
Paper under double-blind review

Abstract

Spatiotemporal graph neural networks (STGNNs) have shown promising results in many domains, from forecasting to epidemiology. However, understanding the dynamics learned by these models and explaining their behaviour is significantly more difficult than for models that deal with static data. Inspired by Koopman theory, which allows a simple description of intricate, nonlinear dynamical systems, we introduce new explainability approaches for temporal graphs. Specifically, we present two methods to interpret the STGNN's decision process and identify the most relevant spatial and temporal patterns in the input for the task at hand. The first relies on dynamic mode decomposition (DMD), a Koopman-inspired dimensionality reduction method. The second relies on sparse identification of nonlinear dynamics (SINDy), a popular method for discovering governing equations of dynamical systems, which we use for the first time as a general tool for explainability. On semi-synthetic dissemination datasets, our methods correctly identify interpretable features such as the times at which infections occur and the infected nodes. We also validate the methods qualitatively on a real-world human motion dataset, where the explanations highlight the body parts most relevant for action recognition.

1 Introduction

Many complex phenomena can be described by the dynamics of items interacting with each other in space and time, leading to complex spatiotemporal relationships that are naturally modelled by temporal graphs (TGs) (Cini et al., 2023b). Examples are roads and junctions in traffic dynamics (Zhang et al., 2020), arms and legs during human motions (Jain et al., 2016), infections during social contacts (Fritz et al., 2022), social interactions during events (Deng et al., 2019), brain activity (Chen et al., 2025), atmospheric events (Marisca et al., 2024), and many more. Graph neural networks (GNNs) have already proved effective on static graphs (Wu et al., 2020; Khemani et al., 2024), but capturing both the spatial and temporal patterns in TGs remains challenging. Recently, spatiotemporal graph neural networks (STGNNs) have emerged as powerful tools to handle this type of data (Longa et al., 2023; Micheli & Tortorella, 2022; Cini et al., 2023a; 2024). Given their use in critical applications, model explainability for STGNNs has become a primary concern (Hassija et al., 2024).

The field of explainability for static GNNs has been very prolific in recent years, offering many methods that can be broadly categorised into factual and counterfactual approaches (Kakkad et al., 2023; Longa et al., 2025). The former approach aims to find important substructures in the input graph, for example, by employing perturbation methods such as GNNExplainer (Ying et al., 2019) or PGExplainer (Luo et al., 2020; Guerra et al., 2023). These are further classified into post hoc methods, which are applied to a trained GNN, and self-interpretable models (Spinelli et al., 2022; 2023; Azzolin et al., 2025b; 2026). Counterfactual methods, instead of identifying the most important subgraph, aim at finding the minimal change in the input graph that causes a change in the GNN's prediction, e.g. CF-GNNExplainer (Lucic et al., 2022). Explainability methods for GNNs can be further classified into local and global approaches: the former, such as the aforementioned PGExplainer, provide an explanation for each input, while global explainers, such as GLGExplainer (Azzolin et al., 2023) or GNNInterpreter (Wang & Shen, 2023), search for common patterns that explain the general behaviour of the model.

Despite preliminary work extending standard explainability techniques to STGNNs for some specific applications in the energy and medical fields (Verdone et al., 2024; Tang et al., 2023; Chen et al., 2025; Altieri et al., 2023), the presence of both spatial and temporal components combined with the black-box nature of neural networks still makes these models particularly difficult to interpret.

A promising direction to address these explainability challenges comes from the field of dynamical systems. In particular, Koopman theory reformulates a complicated nonlinear dynamical system into a simpler linear representation, at the cost of moving to a potentially infinite-dimensional state space. By transforming the nonlinear dynamics into a linear framework, Koopman theory enables the use of robust, data-driven techniques to approximate the system's evolution directly from empirical measurements. This makes the approach particularly useful in many real-world scenarios where the explicit equations of the dynamical system are unknown, but abundant observation data is available. Since deep learning models can be seen as dynamical systems (Han et al., 2024; Gravina et al., 2025), Koopman theory has recently been applied to design interpretable deep learning architectures (Lusch et al., 2018; Mohr et al., 2021) or to perform post hoc analyses (Naiman & Azencot, 2023). Particularly relevant to TGs is the work in (Melnyk et al., 2023; 2020), which is, however, limited to the analysis of metastable states of the human microbiome. In (Shi et al., 2025), inspired by Koopman theory, a new feature propagation mechanism for GNNs is proposed, but its effects on the model's interpretability are not explored, nor is the method extended to STGNNs.

We extend this line of work by studying how Koopman theory can help interpret a spatiotemporal model trained on complex inputs such as TGs. Analysing the model's embeddings with Koopman-inspired techniques such as dynamic mode decomposition (DMD) allows us to recover both the temporal patterns and the subgraphs that have the largest influence on the model's decision-making process. As an additional contribution, we propose using sparse identification of nonlinear dynamics (SINDy) as an explainability method for TGs for the first time. SINDy is a popular algorithm originally introduced to discover governing equations for complex dynamics.

In our experiments, we demonstrate how the proposed methods correctly highlight important features of the input TG. In particular, in dissemination processes, the explanations accurately locate the times at which infections occur and the nodes involved. We further validate the methods qualitatively on a real-world human motion dataset (Microsoft Research Cambridge-12 (MSRC-12)), where the explanations consistently highlight the body parts most relevant for action recognition.

2 Background

2.1 Koopman operator theory

In (Koopman, 1931), Koopman proved how to translate a finite-dimensional nonlinear dynamical system into an infinite-dimensional linear one. Consider a discrete[1] dynamical system on a D-dimensional state space $\mathcal{M}$

$$\boldsymbol{x}_{t+1} = F(\boldsymbol{x}_t), \tag{1}$$

with state $\boldsymbol{x} \in \mathcal{M}$ and flow map $F : \mathcal{M} \to \mathcal{M}$. Let $\varphi : \mathcal{M} \to \mathbb{C}$ be an *observable* in the Hilbert space L^2, i.e. φ is measurable and the Lebesgue integral of the square of the absolute value of φ is finite. Then define the (discrete-time) Koopman operator κ as

$$\kappa\varphi(\boldsymbol{x}_t) = \varphi(F(\boldsymbol{x}_t)) = \varphi(\boldsymbol{x}_{t+1}). \tag{2}$$

The Koopman operator acts on the infinite-dimensional space of observables but has the benefit of being linear:

$$\begin{aligned} \kappa(a\varphi_1(\boldsymbol{x}) + b\varphi_2(\boldsymbol{x})) &= a\varphi_1(F(\boldsymbol{x})) + b\varphi_2(F(\boldsymbol{x})) \\ &= a\kappa\varphi_1(\boldsymbol{x}) + b\kappa\varphi_2(\boldsymbol{x}). \end{aligned} \tag{3}$$

[1]The description can be extended to the continuous case (Mezić, 2021), but our application uses discrete time, so we will focus on the discrete case only.

2.1.1 Dynamic mode decomposition

In some rare cases, it is possible to find a finite-dimensional subspace of L^2, so the Koopman operator, restricted to that subspace, is both finite-dimensional and linear, allowing the well-studied descriptions of linear systems (Mezić, 2021). In general, however, such an invariant subspace does not exist, so a finite-dimensional approximation of κ must be sought instead. This has motivated a growing body of data-driven approaches, founded on Koopman theory, that have increased in popularity (Brunton et al., 2022). The idea is to approximate κ using trajectories $(\boldsymbol{x}_t)_{t=1}^T$ collected from real dynamical systems, reinforced using a library of nonlinear functions, and then use the approximation to simulate and analyse the system.

One of the first classes of algorithms introduced to approximate κ was dynamic mode decomposition (DMD) (Schmid, 2010). It was introduced in fluid dynamics and transport processes to extract relevant information directly from data, without necessarily knowing the governing equation of the dynamics. The link to the Koopman operator was only clarified later (Kutz et al., 2016; Arbabi & Mezić, 2017).

Suppose we have a dynamical system described by (1), from which we collect measurements $\boldsymbol{h}_t = \varphi(\boldsymbol{x}_t) \in \mathbb{R}^F$ at regularly spaced times. We can then build two matrices of snapshots

$$\begin{aligned} \boldsymbol{H} &= (\boldsymbol{h}_1, \boldsymbol{h}_2, \dots, \boldsymbol{h}_{T-1}), \\ \boldsymbol{H}' &= (\boldsymbol{h}_2, \boldsymbol{h}_3, \dots, \boldsymbol{h}_T). \end{aligned} \tag{4}$$

Then the matrix $\boldsymbol{C} \in \mathbb{R}^{F\times F}$ given by $\boldsymbol{H}' \simeq \boldsymbol{C}\boldsymbol{H}$, namely $\boldsymbol{C} = \boldsymbol{H}'\boldsymbol{H}^\dagger$,[2] approximates the Koopman operator (Kutz et al., 2016). Since the matrix $\boldsymbol{C}$ can be large, the DMD algorithm first computes an singular value decomposition (SVD) or principal component analysis (PCA) of the data matrix $\boldsymbol{H}$, before producing a rank-reduced matrix $\boldsymbol{C}$ (Kutz et al., 2016). States $\boldsymbol{h}_t$ can be decomposed onto the basis of eigenvectors of $\boldsymbol{C}$, called *DMD modes*, and their projection $s^{(i)}(t)$ on the i-th mode contains useful information about the dynamics.

The original DMD algorithm described above has been further developed to extend its use and applicability to a wider range of contexts, beyond the fluid dynamics case, and to tackle some of its shortcomings. A review of DMD variations can be found in (Schmid, 2022; Brunton et al., 2022). For implementations in Python, we refer to (Demo et al., 2018; Ichinaga et al., 2024).

2.1.2 Sparse identification of nonlinear dynamics

In the context of discovering and approximating governing equations from data, another approach, alternative to DMD, was introduced under the name sparse identification of nonlinear dynamics (SINDy) in (Brunton et al., 2016). The idea is to approximate the dynamics in (1) with a library of pre-determined nonlinear functions, only a few of which will be relevant (Brunton & Kutz, 2022).

Consider the matrices of snapshots of the system's state $\boldsymbol{x}_t \in \mathbb{R}^D$,

$$\begin{aligned} \boldsymbol{X} &= (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{T-1})^\top \in \mathbb{R}^{(T-1)\times D} \\ \boldsymbol{X}' &= (\boldsymbol{x}_2, \boldsymbol{x}_3, \dots, \boldsymbol{x}_T)^\top \in \mathbb{R}^{(T-1)\times D}, \end{aligned} \tag{5}$$

and a library of J candidate nonlinear functions,

$$\Theta(\boldsymbol{X}) = (\mathbb{I}, \boldsymbol{X}^2, \dots, \sin(\boldsymbol{X}), \dots) \in \mathbb{R}^{(T-1)\times D\times J}, \tag{6}$$

where each function is applied to $\boldsymbol{X}$ element-wise. The dynamical system (1) can now be approximated with

$$\boldsymbol{X}' = \Theta(\boldsymbol{X})\boldsymbol{\xi}, \tag{7}$$

where $\boldsymbol{\xi} \in \mathbb{R}^J$ is a sparse vector, obtained via sparse regression, that selects only a few of the most relevant terms of the library Θ.

[2] Where † is the Moore-Penrose pseudoinverse.

If we write $\boldsymbol{\xi} = (\xi_1, \dots, \xi_J)^\top$, then equation (7) becomes

$$x_{t+1,d} = \sum_{j=1}^{J} \Theta(\boldsymbol{X})_{t,d,j} \xi_j, \tag{8}$$

where each component ξ_j expresses the importance of the j-th nonlinear function in Θ for the system's dynamics. Crucially, the sparsity of $\boldsymbol{\xi}$ and the linear nature of the equation make the identified model directly interpretable: the non-zero coefficients reveal which interactions drive the dynamics, a property we will exploit for explainability in the context of STGNNs.

2.2 Spatiotemporal models

In the present work, we focus on TGs classification task: the whole input TG is processed by a STGNN trained to predict its class y. This section first defines TGs, then describes in detail STGNNs.

2.2.1 Temporal graph

Temporal graphs (TGs) model data whose spatial relations are represented by a graph, and the node labels and topology are both time-dependent. For a formal treatment of TGs and recent reviews of methods and tasks applied to TGs and time series with dependencies expressed as a graph, we refer to (Longa et al., 2023; Cini et al., 2025; 2023b).

In this work, we rely on the following definition of TG.

Definition 2.1 (Discrete time TG)**.** Given a set of N nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$, we define a temporal graph as a sequence of graphs

$$\mathcal{G} := ((\mathcal{V}_t, \mathcal{E}_t))_{t=1}^{T}, \tag{9}$$

with $\mathcal{V}_t \subseteq \mathcal{V}$ and $\mathcal{E}_t \subseteq \mathcal{E}$. Each node $v \in \mathcal{V}_t$ is equipped with a feature vector $\boldsymbol{x}_t^v \in \mathbb{R}^D$.

To simplify the notation, we will not consider features on edges, and the feature vector of the n-th node at time t will be represented by $\boldsymbol{x}_{t,n}$. The adjacency matrix is

$$(\boldsymbol{A}_t)_{n,m} = \begin{cases} 1 & \text{if } (v_n, v_m) \in \mathcal{E}_t \\ 0 & \text{otherwise} \end{cases}, \tag{10}$$

and it varies with time since edges are not guaranteed to exist at all times. We will denote by $N_{\mathcal{G}}$ the number of TGs in a dataset.

2.2.2 Spatiotemporal graph neural network

To process TGs, we use a snapshot-based (Longa et al., 2023), time-and-space (Cini et al., 2025) STGNN, i.e. a deep learning architecture where temporal and spatial processing cannot be factorised into two separate steps.

As representative STGNNs, we consider three different models:

- a graph convolutional recurrent network (GCRN) (Seo et al., 2018),
- a diffusion convolutional recurrent neural network (DCRNN) (Li et al., 2018),
- and a graph WaveNet (GWN) (Wu et al., 2019).

GCRNs and DCRNNs use a spatial encoder at each time step intertwined with a recurrent neural network (RNN) such as a long short-term memory (LSTM), to process temporal dependencies. Let $\mathcal{N}_t(v_n)$ represent

the neighbourhood of the n-th node v_n at time t, ℓ the layer index, and $\mathrm{Enc}(\cdot)$ a spatial encoder. The model encodes an input $\boldsymbol{x}_{t,n}$ into an embedding $\boldsymbol{h}_{t,n,\ell}$ as follows:

$$\begin{aligned}
\boldsymbol{u}_{t,n,\ell} &= \begin{cases} \boldsymbol{x}_{t,n} & \ell = 1 \\ \boldsymbol{h}_{t,n,\ell-1} & \ell > 1 \end{cases}, \\
\boldsymbol{z}_{t,n,\ell} &= \mathrm{Enc}(\{\boldsymbol{u}_{t,m,\ell}\}_{v_m \in \mathcal{N}_t(v_n)}, \boldsymbol{A}_t), \\
\boldsymbol{i}_{t,n,\ell} &= \sigma(\boldsymbol{W}^{\mathrm{zi}} \boldsymbol{z}_{t,n,\ell} + \boldsymbol{W}^{\mathrm{hi}} \boldsymbol{h}_{t-1,n,\ell}), \\
\boldsymbol{f}_{t,n,\ell} &= \sigma(\boldsymbol{W}^{\mathrm{zf}} \boldsymbol{z}_{t,n,\ell} + \boldsymbol{W}^{\mathrm{hf}} \boldsymbol{h}_{t-1,n,\ell}), \\
\boldsymbol{g}_{t,n,\ell} &= \tanh(\boldsymbol{W}^{\mathrm{zg}} \boldsymbol{z}_{t,n,\ell} + \boldsymbol{W}^{\mathrm{hg}} \boldsymbol{h}_{t-1,n,\ell}), \\
\boldsymbol{o}_{t,n,\ell} &= \sigma(\boldsymbol{W}^{\mathrm{zo}} \boldsymbol{z}_{t,n,\ell} + \boldsymbol{W}^{\mathrm{ho}} \boldsymbol{h}_{t-1,n,\ell}), \\
\boldsymbol{c}_{t,n,\ell} &= \boldsymbol{f}_{t,n,\ell} \odot \boldsymbol{c}_{t-1,n,\ell} + \boldsymbol{i}_{t,n,\ell} \odot \boldsymbol{g}_{t,n,\ell}, \\
\boldsymbol{h}_{t,n,\ell} &= \boldsymbol{o}_{t,n,\ell} \odot \tanh(\boldsymbol{c}_{t,n,\ell}).
\end{aligned} \tag{11}$$

GCRNs use a graph convolution network (GCN) (Kipf & Welling, 2017) at each time step as the spatial encoder $\mathrm{Enc}(\cdot)$. DCRNNs use a diffusion convolutional layer (DCL), which simulates, at each time step, a diffusion process with learnable transition probabilities over the graph according to the adjacency matrix $\boldsymbol{A}_t$:

$$\boldsymbol{z}_{t,n,\ell} = \mathrm{DCL}(\{\boldsymbol{u}_{t,m,\ell}\}_{m=1}^{N}) = \sum_{k=0}^{K-1} \sum_{m=1}^{N} (\boldsymbol{P}_t^k)_{n,m} \boldsymbol{u}_{t,m,\ell} \boldsymbol{W}_k, \tag{12}$$

where $\boldsymbol{P}_t = \boldsymbol{D}_t^{-1} \boldsymbol{A}_t$, with $\boldsymbol{D}_t$ the degree matrix, is the transition matrix of the diffusion process, and K is the number of steps.

The third STGNN we consider, GWN, has a different structure. Each layer consists of a (gated) temporal convolution layer (TCN) for the temporal part and a modified DCL for the spatial component:

$$\begin{aligned}
\boldsymbol{z}_{t,n,\ell} &= \tanh(\boldsymbol{W}_1 \star \boldsymbol{u}_{:,n,\ell})_t \odot \sigma(\boldsymbol{W}_2 \star \boldsymbol{u}_{:,n,\ell})_t, \\
\boldsymbol{h}_{t,n,\ell} &= \mathrm{DCL}(\{\boldsymbol{z}_{t,m,\ell}\}_{m=1}^{N}) + \sum_{k=0}^{K} \sum_{m=1}^{N} \tilde{\boldsymbol{A}}_{n,m}^k \boldsymbol{z}_{t,m,\ell} \tilde{\boldsymbol{W}}_k,
\end{aligned} \tag{13}$$

where $\star$ is the dilated causal convolution operation, with dilation d and temporal kernel K_t; to the DCL described in (12), the GWN model adds an extra diffusion term with a trainable time-independent adjacency matrix $\tilde{\boldsymbol{A}}$.

For node-level tasks, the embedding for the n-th node is given by $\boldsymbol{h}_n \in \mathbb{R}^{FL}$, obtained by concatenating, along the layer dimension, the embeddings at the last time step of the STGNN $\boldsymbol{h}_{T,n,\ell} \in \mathbb{R}^F$.[3] On the other hand, the output y of a graph-level task is given by processing with a multi-layer perceptron (MLP) the sum of all node embeddings $\boldsymbol{h} = \sum_{n=1}^{N} \boldsymbol{h}_n \in \mathbb{R}^{FL}$,

$$y = \mathrm{MLP}(\boldsymbol{h}) \in \mathbb{R}^C, \tag{14}$$

where C is the dimension of the desired output, e.g. the number of classes.

For a classification task, the model is trained using a cross-entropy loss between the model's output y and the class label $\hat{y} \in \{1, \dots, C\}$,

$$\ell_{\mathrm{ce}}(y, \hat{y}) = -\log \frac{\exp y_{\hat{y}}}{\sum_{c=1}^{C} \exp y_c}. \tag{15}$$

3 Methods

3.1 Research hypotheses

Providing an instance-based explanation usually involves computing weights that highlight the most relevant parts of the input. In the case of TGs, that means finding a weight $w_t(t)$ for time step t, and spatial weights

[3] Another option is to simply take the last layer embedding $\boldsymbol{h}_n := \boldsymbol{h}_{T,n,L} \in \mathbb{R}^F$.

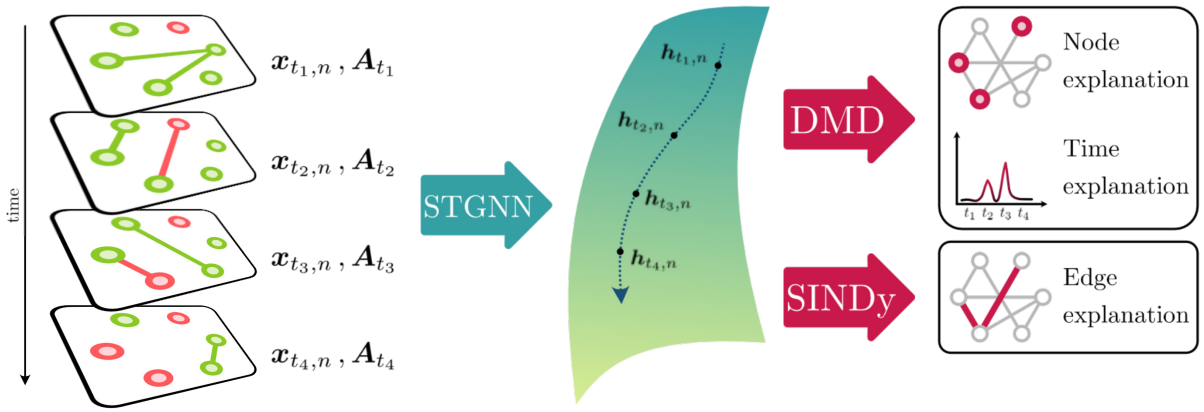


Figure 1: Overview of the proposed framework: the input TG (*left*) is processed via an STGNN, which produces a trajectory of embeddings $\boldsymbol{h}_{t,n}$ (*centre*), whose dynamics is analysed via DMD and SINDy to produce spatial and temporal explanations (*right*).

$w_{\mathtt{s}}(n)$ for node n, or $w_{\mathtt{e}}(n,m)$ for edge (n,m). To properly measure the performance of our explainability methods, we require ground truth for each of these quantities. We call $m_{\mathtt{t}}(t)$ the time ground truth, $m_{\mathtt{s}}(n)$ the spatial ground truth on nodes, and $m_{\mathtt{e}}(n,m)$ the spatial ground truth on edges.

We propose two separate post hoc explainability methods, the first based on DMD and the second on SINDy, that provide either temporal or spatial explanations for TGs, or both. We state below the hypotheses on how DMD and SINDy can explain the STGNN model:

1. The input drives the dynamics of at least one DMD state component $s^{(i)}(t)$ of the TG's embedding, where $s^{(i)}(t)$ is the projection of the STGNN's embedding $\boldsymbol{h}_t$ onto the i-th DMD mode (see Section 3.1.1). Hence, a sudden change in $s^{(i)}(t)$ at time t indicates that the model's internal state reacted to a task-relevant event in the input at that moment. We can therefore use the derivative $ds^{(i)}(t)/dt$ to compute the time weight $w_{\mathtt{t}}(t)$.

2. The projection $s_n^{(i)}(t)$ of a node embedding at each time step (in particular, the last one T) identifies whether that node is important for the output or not, and so it is a proxy for the spatial explanation $w_{\mathtt{s}}(n)$.

3. SINDy fits the dynamics of each node's embedding as a sparse combination of library terms. By construction, the mixed terms in the library correspond to interactions between neighbouring nodes, i.e. edges of the input TG. A large regression weight ξ_j for a term involving edge (n,m) therefore indicates that this interaction strongly drives the embedding dynamics, and hence that the edge is important for the model's prediction. This allows us to use ξ_j to compute the edge explanation $w_{\mathtt{e}}(n,m)$.

3.1.1 Explainability using DMD

We apply DMD to analyse the trajectories of the STGNN's states and understand how inputs are processed by STGNN and affect their output. To reduce the computational complexity of the analysis, a common first step (Naiman & Azencot, 2023) when working with these techniques consists in applying PCA or SVD to reduce the dimensionality of the embeddings from FL to f. An alternative approach is to use tensor-based dynamic mode decomposition (TT-DMD) (Klus et al., 2018; Oseledets, 2011), described in more detail in Appendix A.

Let $\boldsymbol{h}'_{t,n} \in \mathbb{R}^f$ and $\boldsymbol{h}'_t \in \mathbb{R}^f$ be the embeddings of the nodes and the whole TG, respectively, projected onto the principal components. To apply DMD to the whole dataset, we use ridge regression to fit an operator

$\boldsymbol{C} \in \mathbb{R}^{f\times f}$ on the training split of the dataset of TGs embeddings:

$$\boldsymbol{h}'_{t+1} = \boldsymbol{C}\boldsymbol{h}'_t. \tag{16}$$

We can then diagonalise $\boldsymbol{C}$, with eigenvalues $\lambda_i \in \mathbb{C}$ and eigenvectors $\boldsymbol{v}_i \in \mathbb{C}^f$, and study the dynamics along the eigenspaces $\langle \boldsymbol{v}_i \rangle$. We denote by $s^{(i)}(t) = \boldsymbol{v}_i^\top \boldsymbol{h}'_t$ the projection onto the i-th DMD mode at time t (ordered according to the magnitude of the corresponding eigenvalue λ_i). The analysis of $s^{(i)}(t)$ is then performed on the validation set, which allows us to evaluate the generalisation capability of the explainability framework.

On the other hand, to apply DMD on the nodes we focus on a single TG $\mathcal{G}$ at a time, and consider its nodes' embeddings $\boldsymbol{h}'_{t,n}$ to fit a matrix $\boldsymbol{C}_\mathcal{G} \in \mathbb{R}^{f\times f}$

$$\boldsymbol{h}'_{t+1,n} = \boldsymbol{C}_\mathcal{G}\boldsymbol{h}'_{t,n}. \tag{17}$$

As in the previous case, we diagonalise $\boldsymbol{C}_\mathcal{G}$ and compute the projection on the DMD modes for nodes, $s_n^{(i)}(t)$. Unlike before, however, we cannot split the nodes of the TG $\mathcal{G}$ into training and validation sets. Thus, we fit and analyse $\boldsymbol{C}_\mathcal{G}$ on all nodes.

When $|\lambda_i| < 1$, we expect both $s^{(i)}(t)$ and $s_n^{(i)}(t)$ to decay to 0 and contribute little to the final state (see hypotheses 1 and 2). Conversely, when $|\lambda_i| \simeq 1$, the corresponding component $s^{(i)}(t)$ neither grows nor decays, but persists throughout the sequence. In this regime, the component's evolution is not dominated by transient effects and instead tracks the non-trivial spatiotemporal structure of the input, making it a meaningful proxy for the model's decision process. For this reason, we select DMD modes with $|\lambda_i| \simeq 1$ and define the time weight

$$w_\mathtt{t}^{(i)}(t) := \left|\frac{ds^{(i)}(t)}{dt}\right|, \tag{18}$$

and the node weight as the distance from the average,

$$w_\mathtt{s}^{(i)}(n) := \left|s_n^{(i)}(T) - \frac{1}{|\mathcal{V}|}\sum_m s_m^{(i)}(T)\right|. \tag{19}$$

Hypotheses 1 and 2 can be tested by comparing equations (18) and (19) with $m_\mathtt{t}(t)$ and $m_\mathtt{s}(n)$, respectively.

The same approach allows us to define a combined explanation for the entire TG $\mathcal{G}$, which jointly accounts for spatial and temporal patterns. Indeed, we can extend the definition (19) by considering all time steps and not just the last one. We therefore define

$$w_\mathcal{G}^{(i)}(t,n) := \left|s_n^{(i)}(t) - \frac{1}{|\mathcal{V}|}\sum_m s_m^{(i)}(t)\right|, \tag{20}$$

such that $w_\mathtt{s}^{(i)}(n) = w_\mathcal{G}^{(i)}(T,n)$. We refer to Section 4.3 for a discussion on the results obtained with this approach.

3.1.2 Explainability using SINDy

As a data-driven method for approximating governing equations, SINDy can also learn and store useful information about the dynamics of $\boldsymbol{h}'_{t,n}$. In our case, the matrices defined in equations (5) become

$$\begin{aligned}\boldsymbol{H}_n &= (\boldsymbol{h}'_{1,n}, \boldsymbol{h}'_{2,n}, \dots, \boldsymbol{h}'_{T-1,n}) \in \mathbb{R}^{T-1,f}\\ \boldsymbol{H}'_n &= (\boldsymbol{h}'_{2,n}, \boldsymbol{h}'_{3,n}, \dots, \boldsymbol{h}'_{T,n}) \in \mathbb{R}^{T-1,f},\end{aligned} \tag{21}$$

where $n = 1, \dots, N$. The difficult and somewhat arbitrary part of SINDy is the choice of the library of nonlinearities Θ. In our case, though, we can take advantage of this flexibility to enforce a bias in the reconstructed dynamics. Since the GCN and DCL components aggregate information from neighbouring nodes, the next embedding of a node is by construction a nonlinear function of its own and its neighbours'

current embeddings. We can therefore align the SINDy library with this known inductive bias, and restrict the candidate terms to monomials involving the embeddings of neighbouring node pairs. Concretely, the library for node n contains only those terms involving the node itself and its neighbours, namely

$$\begin{aligned}\Theta(\boldsymbol{H}_n) = (&\boldsymbol{H}_n^2, \boldsymbol{H}_n^3, \boldsymbol{H}_n\boldsymbol{H}_{m_1}, \dots, \boldsymbol{H}_n\boldsymbol{H}_{m_1}^2,\\ &\dots, \boldsymbol{H}_n\boldsymbol{H}_{m_2}, \dots, \boldsymbol{H}_n\boldsymbol{H}_{m_2}^2, \dots),\end{aligned} \tag{22}$$

where all operations are performed element-wise, and the node indices m_i refer to nodes that are connected to the n-th node at least once, i.e. all those m_i for which there exists a time t such that $(\boldsymbol{A}_t)_{n,m_i} = 1$. While we could consider higher-order terms, such as $\boldsymbol{H}_n^2\boldsymbol{H}_m^2$, and nodes more than one hop away from n, for the sake of simplicity, we consider monomials up to degree 3 and only one-hop neighbourhoods. We treat the order of the monomials, d_{SINDy}, as a hyperparameter.

After introducing an extra index for the node dimension, equation (8) becomes

$$h'_{t+1,n,d} = \sum_{j=1}^{J} \Theta(\boldsymbol{H}_n)_{t,d,j}\, \xi_{n,j}, \tag{23}$$

where the index $j = 1, \dots, J$ spans the monomials of Θ, each corresponding by construction to an edge of the input TG, including self-loops. We can interpret $\xi_{n,j}$ as a weight that measures how strongly the j-th term contributes to the dynamics of the n-th node embedding, $\boldsymbol{h}'_n$. This allows us to define a weight for each edge (n, m) as

$$w_{\mathsf{e}}(n, m) := \sum_{n'=1}^{N} \sum_{j \sim (n,m)} |\xi_{n',j}|. \tag{24}$$

In equation (24), the inner sum runs over all monomial weights $\xi_{n',j}$ that refer to the same edge (n, m), which expresses how important the edge (n, m) is for the n'-th node. The inner sum is needed because different monomials can refer to the same edge, e.g. the terms $\boldsymbol{H}_n\boldsymbol{H}_m^2$ and $\boldsymbol{H}_n^2\boldsymbol{H}_m$ both relate to (n, m). The outer sum runs over all nodes and therefore measures how important edge (n, m) is for the whole TG. The edge weight $w_{\mathsf{e}}(n, m)$ can then be compared with the ground truth $m_{\mathsf{e}}(n, m)$ to test hypothesis 3.

4 Experiments

To ensure reproducibility, we make our code and experimental setup available in a public repository.[4]

4.1 Datasets

We test the hypotheses above on two types of datasets: the first one consists of semi-synthetic datasets for binary classification tasks with explainability ground truth, while the second is a real-world action classification dataset with no explainability ground truth.

In the first class of datasets, we have TGs whose time-varying topologies describe different types of social interactions. The time-varying node labels $x_{t,n} \in \{0, 1\}$ are produced via a dissemination process simulated with the susceptible-infected model (Oettershagen et al., 2020). TGs of class 1 correspond to dissemination processes. In TGs of class 0, instead, the infected nodes found via the same dissemination process are shuffled randomly at each time step. In each dataset, the two classes are balanced.

While our methods are general and applied to both these semi-synthetic datasets and a real-world dataset (see below), the availability of ground truth in the semi-synthetic datasets allows us to quantitatively measure explainability performance. Specifically, the time ground truth $m_{\mathsf{t}}(t)$ counts the infections occurring at each time step t between adjacent nodes. That is, a time step t contributes to $m_{\mathsf{t}}(t)$ if $x_{t,n}x_{t,m} = 0$ and $x_{t+1,n}x_{t+1,m} = 1$ for some adjacent nodes n, m. The spatial ground truth on nodes $m_{\mathsf{s}}(n)$ indicates which nodes have been infected: $m_{\mathsf{s}}(n) = 1$ if node n has been infected, $m_{\mathsf{s}}(n) = 0$ otherwise. The spatial ground

[4] GitHub repository

truth on edges $m_{\mathtt{e}}(n, m)$ is computed by finding the edges that transmit the infection: $m_{\mathtt{e}}(n, m) = 1$ if there has been an infection between nodes n and m, 0 otherwise.

As mentioned, we also test our approach on the real-world dataset MSRC-12 (Fothergill et al., 2012), consisting of sequences of 12 human movements, where the node labels $\boldsymbol{x}_{t,n} \in \mathbb{R}^3$ represent the 3D joint coordinates. When training a STGNN on this dataset, each input is augmented by performing random rotations in 3D space. Unlike the semi-synthetic datasets, in MSRC-12 the topology does not change over time, and there is no explainability ground truth, so we provide only qualitative results for our explainability methods. Moreover, since the sequences have different lengths, each one needs to be padded along the time dimension before forming batches.

See Appendix B for further details on each dataset.

4.2 Metrics

For those datasets that come with a ground truth, we can define metrics to assess quantitatively whether the hypotheses formulated in Section 3.1 hold. To test hypothesis 1, we measure the agreement between the time ground truth $m_{\mathtt{t}}(t)$ and the time weight $w_{\mathtt{t}}^{(i)}(t)$. Projections on DMD modes can be quite noisy, while the time ground truth $m_{\mathtt{t}}(t)$ is very sharp, being 0 almost everywhere and positive only at a few sparse time steps. This poses a significant challenge when comparing two time signals, common in fields such as anomaly detection (Wagner et al., 2026; Kim et al., 2022). To overcome this, we consider a regularised version of the time ground truth, obtained by convolving $m_{\mathtt{t}}(t)$ with a uniform filter to make it smoother before computing the F1 metric, and we apply a threshold in different ways:

- *F1 with thresholds.* The F1 score between the time ground truth and the time steps t such that $w_{\mathtt{t}}^{(i)}(t) > \delta$, with δ being a threshold:

  MAX $\delta = \delta' \cdot \max_t w_{\mathtt{t}}^{(i)}(t)$, i.e. the threshold is a fraction of the maximum value of $w_{\mathtt{t}}^{(i)}(t)$;
  AVG $\delta = \mu_{w_{\mathtt{t}}} + \sigma_{w_{\mathtt{t}}}$, where $\mu_{w_{\mathtt{t}}}$ and $\sigma_{w_{\mathtt{t}}}$ are the time average and standard deviation respectively;
  MAD $\delta = \bar{w}_{\mathtt{t}} + k \cdot \text{median}(|w_{\mathtt{t}} - \bar{w}_{\mathtt{t}}|)$, where k is a hyperparameter (we use $k = 3$), and $\bar{w}_{\mathtt{t}} = \text{median}(w_{\mathtt{t}})$.

- *F1 with window average.* As above, but in addition, we first take a running average of $w_{\mathtt{t}}^{(i)}(t)$ to reduce noise, with window size ω.

For hypothesis 2, we compare $w_{\mathtt{s}}^{(i)}(n)$ from equation (19) with the explanation ground truth $m_{\mathtt{s}}(n)$. Since identifying the explanation is a binary classification problem at the node level, we can measure the area under the curve (AUC) score between $w_{\mathtt{s}}^{(i)}(n)$ and $m_{\mathtt{s}}(n)$, a metric often called *plausibility* in the literature (Longa et al., 2025). We denote it $\text{AUC}_{\mathcal{G}}$. We refer to (Fontanesi et al., 2025) for a discussion on the challenges of explaining GNNs even in the presence of a ground truth.

For hypothesis 3, we use weights $w_{\mathtt{e}}(n, m)$ from equation (24), and compare them with $m_{\mathtt{e}}(n, m)$ via an AUC score, called AUC_{edge}.

Standard explainability metrics for GNNs in the absence of ground truth do exist — e.g. *faithfulness*, *fidelity*, and *comprehensiveness* (Agarwal et al., 2023; Fontanesi et al., 2024; Longa et al., 2025; Azzolin et al., 2025a) —, but their extension to the TGs domain is not trivial and remains underexplored (see Dileo et al. (2025) for an early attempt in the context of link prediction). Although we acknowledge that a systematic adaptation of these metrics to temporal graphs is important, it lies beyond the scope of this work; therefore, we leave it as a direction for future research. For this reason, the performance of the proposed methods on the MSRC-12 dataset is explored only qualitatively.

4.3 Results

Before testing the proposed explainability tools, we tune the STGNN hyperparameters to maximise classification accuracy. The accuracies obtained with the best hyperparameters are reported in Table 1.

Table 1: Accuracies of the best-performing STGNNs, averaged over 5 runs.

STGNN	Facebook	Infectious	DBLP	Highschool	Tumblr	MSRC-12
GCRN	0.95 ± 0.02	0.97 ± 0.03	0.991 ± 0.008	0.87 ± 0.11	0.96 ± 0.01	0.87 ± 0.23
DCRNN	0.95 ± 0.01	0.93 ± 0.07	0.988 ± 0.005	0.96 ± 0.06	0.85 ± 0.09	0.963 ± 0.009
GWN	0.95 ± 0.01	0.86 ± 0.04	0.95 ± 0.01	0.89 ± 0.04	0.96 ± 0.02	0.909 ± 0.008

Table 2: Results of experiments to test hypothesis 1. The averages and standard deviations are computed over 5 runs. Methods scoring the highest mean value are reported in **bold**.

	Metrics	Facebook	Infectious	DBLP	Highschool	Tumblr
GCRN	F1	$\mathbf{0.33 \pm 0.03}$	$\mathbf{0.43 \pm 0.14}$	$\mathbf{0.59 \pm 0.13}$	$\mathbf{0.33 \pm 0.15}$	0.23 ± 0.03
	F1naïf	0.004 ± 0.001	0.023 ± 0.008	0.0006 ± 0.0006	0.017 ± 0.008	0.010 ± 0.002
	F1sal	0.28 ± 0.06	0.19 ± 0.11	0.54 ± 0.22	0.21 ± 0.13	$\mathbf{0.37 \pm 0.09}$
DCRNN	F1	0.30 ± 0.06	$\mathbf{0.54 \pm 0.03}$	0.47 ± 0.16	0.41 ± 0.09	0.26 ± 0.06
	F1naïf	0.003 ± 0.001	0.02 ± 0.01	0.0006 ± 0.0006	0.02 ± 0.01	0.010 ± 0.002
	F1sal	$\mathbf{0.45 \pm 0.02}$	0.07 ± 0.03	$\mathbf{0.65 \pm 0.13}$	$\mathbf{0.49 \pm 0.11}$	$\mathbf{0.38 \pm 0.02}$
GWN	F1	0.36 ± 0.02	$\mathbf{0.33 \pm 0.04}$	$\mathbf{0.51 \pm 0.05}$	$\mathbf{0.35 \pm 0.10}$	0.31 ± 0.02
	F1naïf	0.019 ± 0.005	0.29 ± 0.06	0.0006 ± 0.0006	0	0.043 ± 0.007
	F1sal	$\mathbf{0.59 \pm 0.02}$	0.27 ± 0.05	0.44 ± 0.03	0.19 ± 0.05	$\mathbf{0.55 \pm 0.01}$

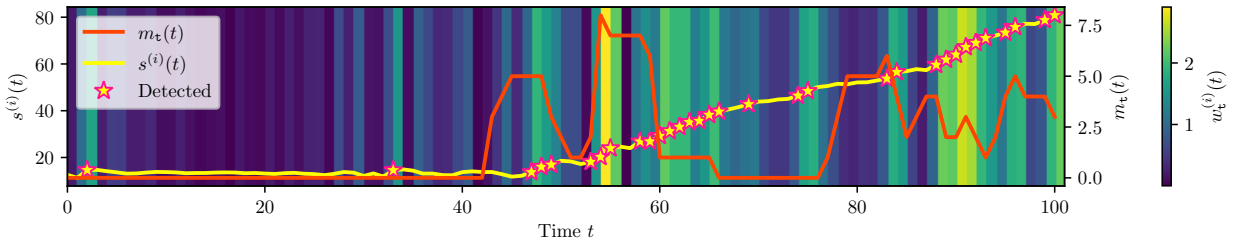
Since the F1 scores depend on the threshold δ and the window size ω, we perform a grid search on these parameters too. We refer to Appendix C for details.

Time explanations To test hypothesis 1, we report in Table 2 the F1 scores comparing the time weights $w_{\mathtt{t}}^{(i)}$ with the time ground truth $m_{\mathtt{t}}$. We also report two baseline values. The first, F1naïf, is obtained from a naïf explainer that outputs $w_{\mathtt{t}}^{(i)}(t) = 1$ for all t. As expected, F1naïf is always very low, evidencing the difficulty of comparing this type of signal (Wagner et al., 2026; Kim et al., 2022). The second baseline, F1sal, is computed using a saliency map as explanation (more details are given in Appendix D) (Simonyan et al., 2014). The F1 scores found with our methods are comparable to or better than those found with saliency. In Figure 2, we report two examples of time explanations from the Facebook dataset: in the top panel, the detection is obtained by thresholding $w_{\mathtt{t}}^{(i)}(t)$ directly, while in the bottom panel we first apply a window average to $w_{\mathtt{t}}^{(i)}(t)$.

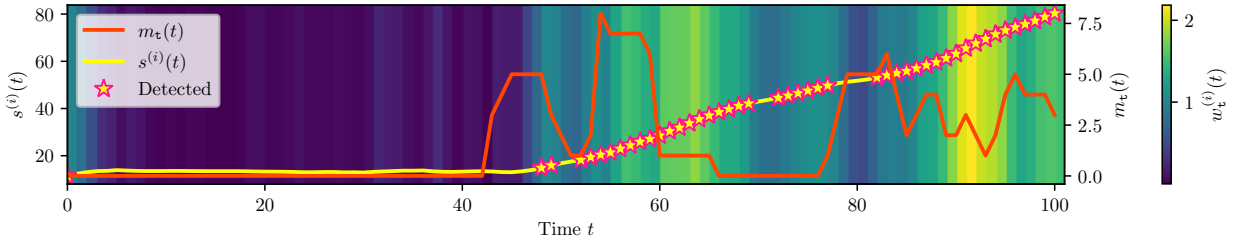
Results associated with the GWN highlight some limitations of our explainability method. As discussed in Section 2.2.2, equation (13), a GWN uses a dilated causal convolution operation with dilation d and temporal kernel $K_{\mathtt{t}}$, which makes the time sequence of embeddings $(\boldsymbol{h}_{t,n})_{t=1}^{T_{\text{out}}}$ shorter than the time sequence of inputs $(\boldsymbol{x}_{t,n})_{t=1}^{T_{\text{in}}}$. The relationship between T_{out} and T_{in},

$$T_{\text{out}} = T_{\text{in}} - (K_{\mathtt{t}} - 1) \sum_{\ell=0}^{L-1} d^{(\ell \bmod 2)}, \tag{25}$$

depending on the choice of hyperparameters, can significantly affect the length of the sequence of states to which we apply our methods, especially if the dataset consists of short input sequences. In choosing the hyperparameters, we traded off some accuracy to prevent T_{out} from becoming too short, which in turn negatively affects the effectiveness of the explanations. This is particularly apparent for the Infectious dataset.

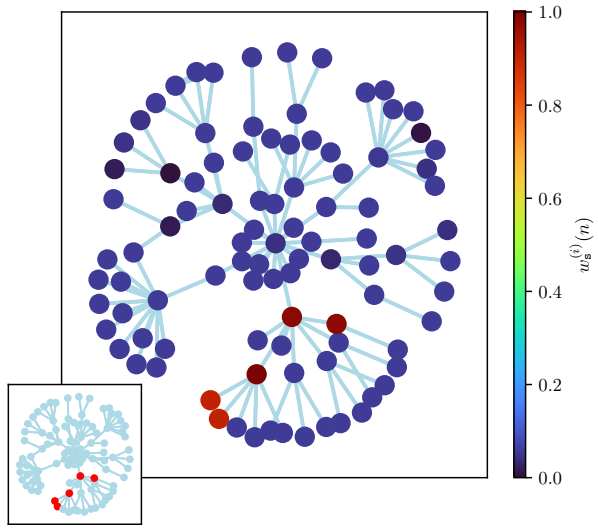


(a) Time explanation via threshold. The F1 score is 0.68, affected by some false negatives.

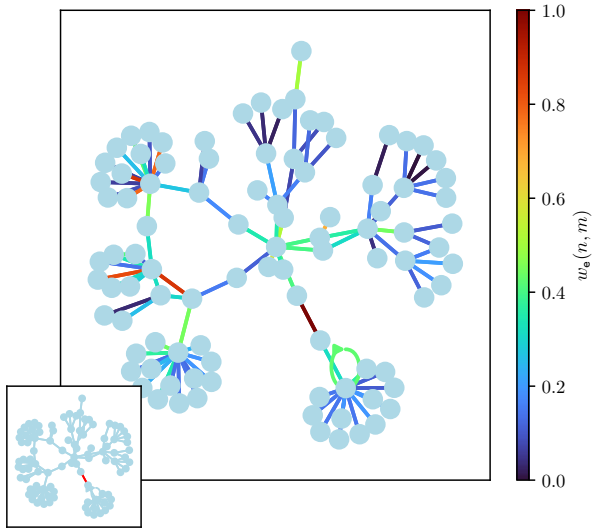


(b) Time explanation via window average. The F1 score is 0.81.

Figure 2: Examples of time explanations for the Facebook dataset and GCRN model. The red line represents the smoothed ground truth $m_t(t)$, the yellow line is the relevant component $s^{(i)}(t)$, the background colour scale shows the explanation weight $w_t^{(i)}(t)$, the stars highlight those times t where $w_t^{(i)}(t) > \delta$.



(a) Node mask with $\text{AUC}_{\mathcal{G}} = 1$.



(b) Edges mask with $\text{AUC}_{\text{edge}} = 1$.

Figure 3: Spatial explanations from the Facebook dataset. The colour scale on the nodes or edges represents the explanation weights $w_s^{(i)}(n)$ (for nodes) and $w_e(n, m)$ (for edges). The ground truth is reported in the corner.

Spatial explanations To evaluate hypotheses 2 and 3, we report the AUC scores in Table 3. We also report the AUC scores of the node explanations provided by the saliency map (see Appendix D for more details). All proposed methods that provide spatial explanations perform consistently well, with some differences across datasets. Even when compared with saliency, our methods almost always perform better

Table 3: Results of experiments to test hypotheses 2 and 3. The averages and standard deviations are computed over 5 runs.

	Metrics	Facebook	Infectious	DBLP	Highschool	Tumblr
GCRN	AUC_{edge}	0.923 ± 0.002	0.71 ± 0.08	0.85 ± 0.03	0.73 ± 0.09	0.72 ± 0.04
	$\text{AUC}_{\mathcal{G}}$	0.849 ± 0.005	0.74 ± 0.09	0.63 ± 0.03	0.66 ± 0.20	0.88 ± 0.02
	AUC_{sal}	0.44 ± 0.02	0.50 ± 0.25	0.81 ± 0.14	0.52 ± 0.17	0.89 ± 0.05
DCRNN	AUC_{edge}	0.84 ± 0.01	0.66 ± 0.05	0.83 ± 0.02	0.67 ± 0.10	0.81 ± 0.05
	$\text{AUC}_{\mathcal{G}}$	0.859 ± 0.003	0.67 ± 0.01	0.72 ± 0.04	0.62 ± 0.20	0.80 ± 0.09
	AUC_{sal}	0.36 ± 0.02	0.67 ± 0.37	0.16 ± 0.09	0.43 ± 0.11	0.82 ± 0.09
GWN	AUC_{edge}	0.77 ± 0.03	0.59 ± 0.02	0.70 ± 0.04	0.61 ± 0.05	0.68 ± 0.04
	$\text{AUC}_{\mathcal{G}}$	0.83 ± 0.02	0.69 ± 0.05	0.82 ± 0.03	0.56 ± 0.11	0.73 ± 0.04
	AUC_{sal}	0.15 ± 0.07	0.37 ± 0.30	0.25 ± 0.24	0.12 ± 0.06	0.22 ± 0.07

than the baseline, demonstrating their effectiveness. In Figure 3, we report an example of an explanation on both nodes and edges.

Although the analysis is instance-based, we can leverage our methods to infer something more general about the model's behaviour. For example, we notice that the weight $w_{\mathcal{G}}^{(i)}(t, n)$ effectively recognises whether the n-th node is infected or not at time t. This means that the quantity $\sum_n w_{\mathcal{G}}^{(i)}(t, n)$ is proportional to the number of infected nodes at each time step t. In other words, it reveals a behaviour of the STGNN that transcends the specific input, namely that it learns to count infected nodes. Even though this information alone is not sufficient to tell the two classes apart, it is an implicit feature that emerges as the model learns to solve the task at hand. Therefore, we argue that the proposed tools can also help interpret model behaviour, not only the input data.

Combined spatiotemporal explanations We can combine the two approaches and use the spatiotemporal weight $w_{\mathcal{G}}^{(i)}(t, n)$ defined in (20), comparing it with a spatiotemporal ground truth $m_{\mathtt{st}}(t, n)$.

To assess the agreement with the ground truth qualitatively, we refer to Figure 4, which shows an example of a spatiotemporal explanation for the GCRN model from the Facebook dataset. The colour scale in the background represents $w_{\mathcal{G}}^{(i)}(t, n)$, and the red boxes indicate the ground truth $m_{\mathtt{st}}^{(i)}(t, n)$.

To provide a more quantitative measure of the agreement between the spatiotemporal explanation $w_{\mathcal{G}}^{(i)}(t, n)$ and the mask $m_{\mathtt{st}}(t, n)$, one option is to use the Brier score, defined as

$$\text{BS}(t) := \frac{1}{|\mathcal{V}|} \sum_{n=1}^{|\mathcal{V}|} \left(\frac{w_{\mathcal{G}}^{(i)}(t, n)}{\max(w_{\mathcal{G}}^{(i)}(t, n))} - m_{\mathtt{st}}(t, n) \right)^2 . \tag{26}$$

We choose the Brier score to measure accuracy because it correctly accounts for imbalanced classes and it also provides an easily interpretable outcome, where $\text{BS}(t) = 0$ is the best value and $\text{BS}(t) = 1$ is the worst. The Brier score is depicted at the bottom of Figure 4: the bumps in the plot correspond to the region with more disagreement between the prediction and the ground truth. In particular, we notice that there is a delay before the explanation registers the infection of a node, and two nodes are false positives, but the Brier score is consistently close to 0.

Qualitative explanations For the MSRC-12 dataset, given the lack of a ground truth (see the discussion in Section 4.2), we rely on a qualitative analysis of the explanations provided by our methods. In Figures 5 to 7, we plot 10 frames sampled from three sequences, representing the actions "change weapon", "take a bow" and "crouch", respectively. The colour scale on the left represents the node weight $w_{\mathcal{G}}(t, n)$ from

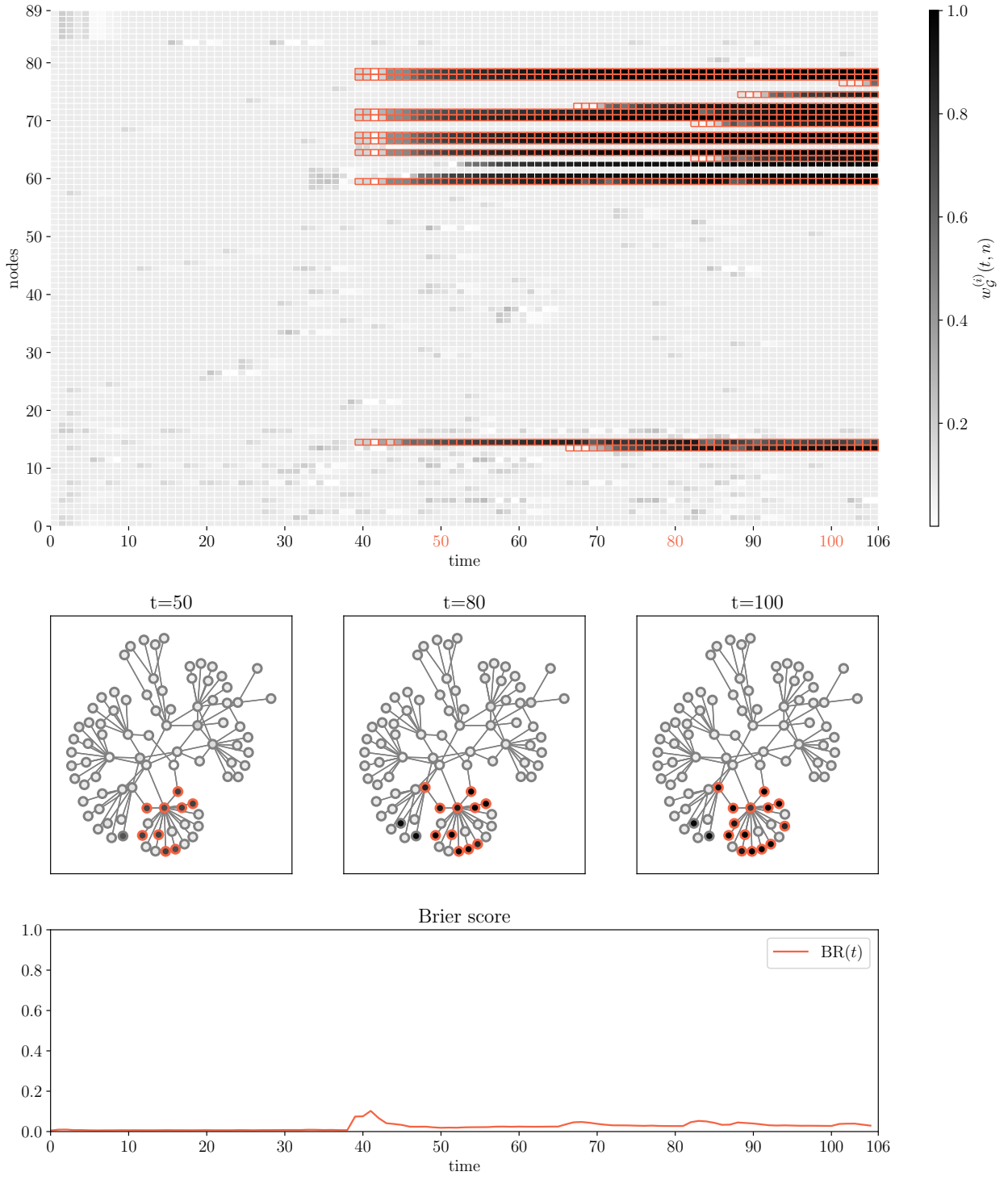


Figure 4: *Top*: The colour scale represents the explanation $w_{\mathcal{G}}^{(i)}(t,n)$ for each time step (x axis) and each node (y axis). The red squares mark the entries for which $m_{\mathtt{st}}(t,n) = 1$. *Middle*: The panel shows the TG $\mathcal{G}$ at three times, $t = 50,\ 80,\ 100$. Nodes in the ground truth are highlighted in red. *Bottom*: The panel shows the value of the Brier score $\mathrm{BS}(t)$.

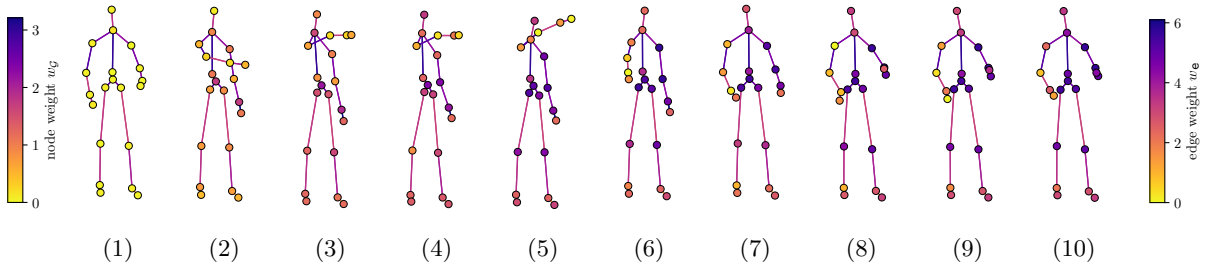


Figure 5: One example from MSRC-12 dataset, corresponding to class “change weapon”.

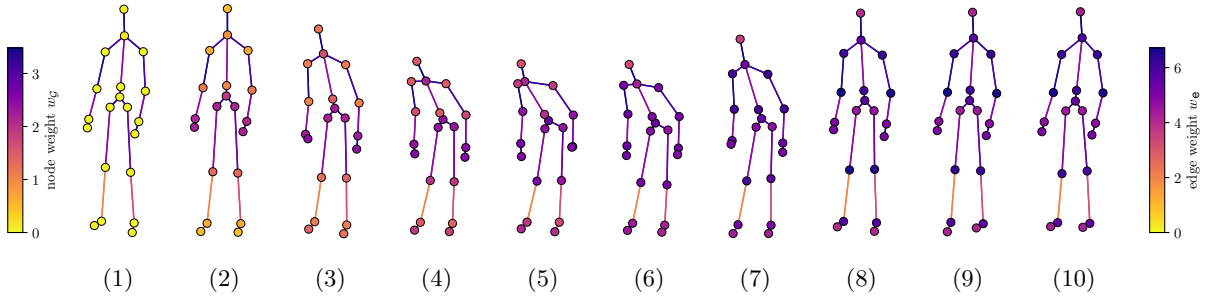


Figure 6: One example from MSRC-12 dataset, corresponding to class “take a bow”.

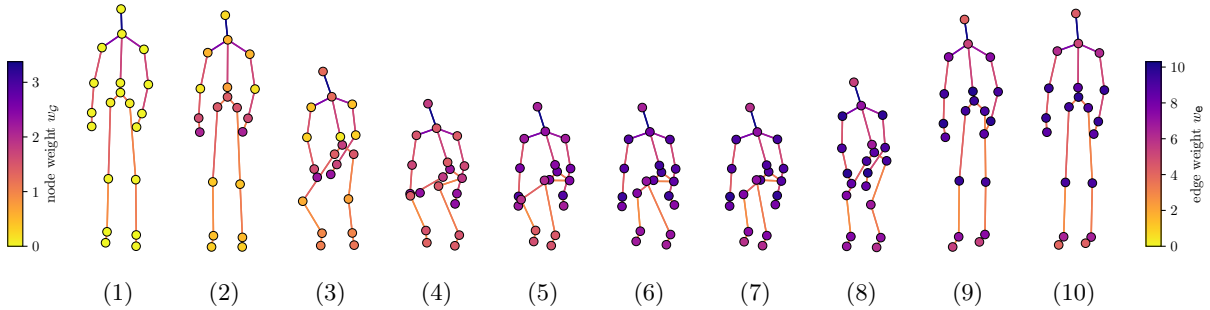


Figure 7: One example from MSRC-12 dataset, corresponding to class “crouch”.

equation (20), while the colour scale on the right represents the edge weight $w_{\mathsf{e}}(n, m)$ from equation (24), whose value is constant in time. For these particular examples, we use GCRN as the model, TT-DMD to compute $w_{\mathcal{G}}(t, n)$, and SINDy with degree 3 to compute $w_{\mathsf{e}}(n, m)$. All the reported instances are correctly classified by the model. Other examples from the remaining classes are reported in Appendix E.

Here are some qualitative comments on the figures:

- *Class "change weapon"*. In Figure 5, we see that the movement involves only the arms: one reaches behind the back to pick up the weapon, and the other holds it. The node explanation shows that the model focuses more on the arm on the right side of the figure, which is consistent with what we see in other samples of the same class. One possible explanation is that the movement of a single arm provides enough discriminative power, given that other classes, e.g. "protest the music", also involve arm movements only. The pelvis nodes are also important to the model because of their rotational motion. The edge weights mainly highlight the arms as well.

- *Class "take a bow"*. In Figure 6, the explanation is less localised, with more importance given to the upper body and the knees, both in terms of nodes and edges. This makes sense, since the bowing movement involves almost all nodes: other classes involving the movement of most of the body (e.g. "crouch" and "kick") have more localised explanations (see Figures 7 and 16), so focusing extendendly on more nodes holds enough discriminative power.

- *Class "crouch"*. In Figure 7, the model focuses on the lumbar region, the hips and the knees; the arms seem to play a role too, because in other samples in this class the subjects bend their arms and rest their hands on the knees, although that is not the case in this specific instance. The legs have the highest scores in the edge explanation.

5 Conclusion

In this work, we introduced a Koopman-theoretic perspective on explainability for STGNNs. By treating the internal embeddings of the model as observables of an underlying dynamical system, we showed that data-driven tools from dynamical systems, namely DMD and SINDy, can reveal when relevant events occur, which nodes are most responsible for the prediction, and which interactions are the most influential. The experiments on semi-synthetic dissemination datasets, together with the qualitative analysis on MSRC-12, indicate that this perspective is effective across different STGNN architectures and can recover meaningful temporal, spatial, and edge-level explanations.

A key takeaway is that, although STGNNs are nonlinear models operating on highly structured inputs, their learned latent dynamics still contain enough regularity for Koopman-inspired analyses to be informative. This suggests that explainability for temporal graph models can benefit from dynamical-systems tools, and not only from perturbation-based or saliency-based approaches. More broadly, the proposed framework offers a way to connect the internal representations of STGNNs with interpretable phenomena in the input domain, which may be especially relevant in scientific applications where understanding the evolution of the system is as important as obtaining an accurate prediction.

Several directions remain open. First, the field would benefit from more principled evaluation protocols for explainability on TGs, especially in settings without explanation ground truth. Second, it would be natural to extend this perspective beyond classification to tasks such as forecasting and link prediction, and to study whether the same dynamical structures remain explanatory there. Third, understanding how dynamical priors should be incorporated into model design and training remains an open question, since the relationship between more structured latent dynamics and better explanations is not yet fully understood. We hope that this work will encourage further interaction between Koopman theory, system identification, and explainability for graph-based temporal learning.

References

Chirag Agarwal, Owen Queen, Himabindu Lakkaraju, and Marinka Zitnik. Evaluating explainability for graph neural networks. *Scientific Data*, 10(1):144, 2023.

Massimiliano Altieri, Michelangelo Ceci, and Roberto Corizzo. Explainable spatio-temporal graph modeling. In *International Conference on Discovery Science*, pp. 174–188. Springer, 2023.

Hassan Arbabi and Igor Mezić. Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the koopman operator. *SIAM Journal on Applied Dynamical Systems*, 16(4):2096–2126, 2017.

Steve Azzolin, Antonio Longa, Pietro Barbiero, Pietro Lio, and Andrea Passerini. Global explainability of GNNs via logic combination of learned concepts. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=OTbRTIY4YS.

Steve Azzolin, Antonio Longa, Stefano Teso, and Andrea Passerini. Reconsidering faithfulness in regular, self-explainable and domain invariant GNNs. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=kiOxNsrpQy.

Steve Azzolin, Sagar Malhotra, Andrea Passerini, and Stefano Teso. Beyond topological self-explainable GNNs: A formal explainability perspective. In *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=mkqcUWBykZ.

Steve Azzolin, Stefano Teso, Bruno Lepri, Andrea Passerini, and Sagar Malhotra. GNN explanations that do not explain and how to find them. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=HBcgLe6NZD.

Filippo Maria Bianchi, Simone Scardapane, Sigurd Løkse, and Robert Jenssen. Reservoir computing approaches for representation and classification of multivariate time series. *IEEE Transactions on Neural Networks and Learning Systems*, 32(5):2169–2179, 2021.

Steven L. Brunton and J. Nathan Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2 edition, 2022.

Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113 (15):3932–3937, 2016.

Steven L. Brunton, Marko Budišić, Eurika Kaiser, and J. Nathan Kutz. Modern koopman theory for dynamical systems. *SIAM Review*, 64(2):229–340, 2022.

Longyun Chen, Yuhui Yang, Aiju Yu, Shuo Guo, Kai Ren, Qinfang Liu, and Chen Qiao. An explainable spatio-temporal graph convolutional network for the biomarkers identification of adhd. *Biomedical Signal Processing and Control*, 99:106913, 2025.

Andrea Cini, Ivan Marisca, Filippo Maria Bianchi, and Cesare Alippi. Scalable spatiotemporal graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 7218–7226, 2023a.

Andrea Cini, Daniele Zambon, and Cesare Alippi. Sparse graph learning from spatiotemporal time series. *Journal of Machine Learning Research*, 24(242):1–36, 2023b.

Andrea Cini, Danilo Mandic, and Cesare Alippi. Graph-based Time Series Clustering for End-to-End Hierarchical Forecasting. *International Conference on Machine Learning*, 2024.

Andrea Cini, Ivan Marisca, Daniele Zambon, and Cesare Alippi. Graph deep learning for time series forecasting. *ACM Comput. Surv.*, 57(12), July 2025. ISSN 0360-0300. doi: 10.1145/3742784. URL https://doi.org/10.1145/3742784.

Nicola Demo, Marco Tezzele, and Gianluigi Rozza. Pydmd: Python dynamic mode decomposition. *Journal of Open Source Software*, 3(22):530, 2018.

Songgaojun Deng, Huzefa Rangwala, and Yue Ning. Learning dynamic context graphs for predicting social events. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1007–1016, 2019.

Manuel Dileo, Matteo Zignani, and Sabrina Tiziana Gaito. Evaluating explainability techniques on discrete-time graph neural networks. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=JzmXo0rfry.

Michele Fontanesi, Alessio Micheli, and Marco Podda. Explaining graph classifiers by unsupervised node relevance attribution. In *World Conference on Explainable Artificial Intelligence*, pp. 63–74. Springer, 2024.

Michele Fontanesi, Alessio Micheli, Marco Podda, and Domenico Tortorella. Bridging xai and spectral analysis to investigate the inductive biases of deep graph networks. *Machine Learning*, 114(11):257, 2025.

Simon Fothergill, Helena Mentis, Pushmeet Kohli, and Sebastian Nowozin. Instructing people for training gestural interactive systems. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pp. 1737–1746, 2012.

Cornelius Fritz, Emilio Dorigatti, and David Rügamer. Combining graph neural networks and spatio-temporal disease models to improve the prediction of weekly covid-19 cases in germany. *Scientific Reports*, 12(1):3930, 2022.

Alessio Gravina, Moshe Eliasof, Claudio Gallicchio, Davide Bacciu, and Carola-Bibiane Schönlieb. On oversquashing in graph neural networks through the lens of dynamical systems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(16):16906–16914, Apr. 2025. doi: 10.1609/aaai.v39i16.33858. URL https://ojs.aaai.org/index.php/AAAI/article/view/33858.

Michele Guerra, Indro Spinelli, Simone Scardapane, and Filippo Maria Bianchi. Explainability in subgraphs-enhanced graph neural networks. In *Proceedings of the Northern Lights Deep Learning Workshop*, volume 4, 2023.

Andi Han, Dai Shi, Lequan Lin, and Junbin Gao. From continuous dynamics to graph neural networks: Neural diffusion and beyond. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=fPQSxjqa2o. Survey Certification.

Vikas Hassija, Vinay Chamola, Atmesh Mahapatra, Abhinandan Singal, Divyansh Goel, Kaizhu Huang, Simone Scardapane, Indro Spinelli, Mufti Mahmud, and Amir Hussain. Interpreting black-box models: a review on explainable artificial intelligence. *Cognitive Computation*, 16(1):45–74, 2024.

Sara M Ichinaga, Francesco Andreuzzi, Nicola Demo, Marco Tezzele, Karl Lapo, Gianluigi Rozza, Steven L Brunton, and J Nathan Kutz. Pydmd: A python package for robust dynamic mode decomposition. *Journal of Machine Learning Research*, 25(417):1–9, 2024.

Lorenzo Isella, Juliette Stehlé, Alain Barrat, Ciro Cattuto, Jean-François Pinton, and Wouter Van den Broeck. What's in a crowd? analysis of face-to-face behavioral networks. *Journal of theoretical biology*, 271(1):166–180, 2011.

Ashesh Jain, Amir R. Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pp. 5308–5317, 2016.

Jaykumar Kakkad, Jaspal Jannu, Kartik Sharma, Charu Aggarwal, and Sourav Medya. A survey on explainability of graph neural networks. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 2023.

Bharti Khemani, Shruti Patil, Ketan Kotecha, and Sudeep Tanwar. A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions. *Journal of Big Data*, 11(1):18, 2024.

Siwon Kim, Kukjin Choi, Hyun-Soo Choi, Byunghan Lee, and Sungroh Yoon. Towards a rigorous evaluation of time-series anomaly detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7): 7194–7201, Jun. 2022. doi: 10.1609/aaai.v36i7.20680. URL https://ojs.aaai.org/index.php/AAAI/article/view/20680.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Stefan Klus, Patrick Gelß, Sebastian Peitz, and Christof Schütte. Tensor-based dynamic mode decomposition. *Nonlinearity*, 31(7):3359–3380, 2018.

Bernard O. Koopman. Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931.

J. Nathan Kutz, Steven L. Brunton, Bingni W. Brunton, and Joshua L. Proctor. *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

Jure Leskovec, Lars Backstrom, and Jon Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 497–506, 2009.

Qianxiao Li, Felix Dietrich, Erik M. Bollt, and Ioannis G. Kevrekidis. Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the koopman operator. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(10), 2017.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SJiHXGWAZ.

Antonio Longa, Veronica Lachi, Gabriele Santin, Monica Bianchini, Bruno Lepri, Pietro Lio, Franco Scarselli, and Andrea Passerini. Graph neural networks for temporal graphs: State of the art, open challenges, and opportunities. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

Antonio Longa, Steve Azzolin, Gabriele Santin, Giulia Cencetti, Pietro Liò, Bruno Lepri, and Andrea Passerini. Explaining the explainers in graph neural networks: a comparative study. *ACM Computing Surveys*, 57(5):1–37, 2025.

Ana Lucic, Maartje A Ter Hoeve, Gabriele Tolomei, Maarten De Rijke, and Fabrizio Silvestri. Cf-gnnexplainer: Counterfactual explanations for graph neural networks. In *International Conference on Artificial Intelligence and Statistics*, pp. 4499–4511. PMLR, 2022.

Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *Advances in neural information processing systems*, 33: 19620–19631, 2020.

Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):4950, 2018.

Ivan Marisca, Cesare Alippi, and Filippo Maria Bianchi. Graph-based forecasting with missing data through spatiotemporal downsampling. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 34846–34865. PMLR, 2024.

Kateryna Melnyk, Stefan Klus, Grégoire Montavon, and Tim O.F. Conrad. Graphkke: graph kernel koopman embedding for human microbiome analysis. *Applied Network Science*, 5:1–22, 2020.

Kateryna Melnyk, Kuba Weimann, and Tim O.F. Conrad. Understanding microbiome dynamics via interpretable graph representation learning. *Scientific Reports*, 13(1):2058, 2023.

Igor Mezić. Koopman operator, geometry, and learning of dynamical systems. *Not. Am. Math. Soc.*, 68(7): 1087–1105, 2021.

Alessio Micheli and Domenico Tortorella. Discrete-time dynamic graph echo state networks. *Neurocomputing*, 496:85–95, 2022.

Ryan Mohr, Maria Fonoberova, Iva Manojlović, Aleksandr Andrejčuk, Zlatko Drmač, Yannis Kevrekidis, and Igor Mezić. Applications of koopman mode analysis to neural networks. In *Proceedings of the AAAI 2021 Spring Symposium on Combining Artificial Intelligence and Machine Learning with Physical Sciences*. Aachen: CEUR, 2021.

Ilan Naiman and Omri Azencot. An operator theoretic approach for analyzing sequence neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 9268–9276, 2023.

Lutz Oettershagen, Nils M. Kriege, Christopher Morris, and Petra Mutzel. *Temporal Graph Kernels for Classifying Dissemination Processes*, pp. 496–504. Society for Industrial and Applied Mathematics (SIAM), 2020.

Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

Peter J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.

Peter J. Schmid. Dynamic mode decomposition and its variants. *Annual Review of Fluid Mechanics*, 54(1): 225–254, 2022.

Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. Structured sequence modeling with graph convolutional recurrent networks. In *Neural Information Processing: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part I 25*, pp. 362–373. Springer, 2018.

Dai Shi, Lequan Lin, Andi Han, Zhiyong Wang, Yi Guo, and Junbin Gao. When graph neural networks meet dynamic mode decomposition. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=duGygkA3QR.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop at International Conference on Learning Representations*, 2014.

Indro Spinelli, Simone Scardapane, and Aurelio Uncini. A meta-learning approach for training explainable graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4):4647–4655, 2022.

Indro Spinelli, Michele Guerra, Filippo Maria Bianchi, and Simone Scardapane. Combining stochastic explainers and subgraph neural networks can increase expressivity and interpretability. In *ESANN 2023 proceedings*, pp. 229–234, 01 2023. doi: 10.14428/esann/2023.ES2023-13.

Jiabin Tang, Lianghao Xia, and Chao Huang. Explainable spatio-temporal graph neural networks. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 2432–2441, 2023.

Alessio Verdone, Simone Scardapane, and Massimo Panella. Explainable spatio-temporal graph neural networks for multi-site photovoltaic energy production. *Applied Energy*, 353:122151, 2024.

Bimal Viswanath, Alan Mislove, Meeyoung Cha, and Krishna P. Gummadi. On the evolution of user interaction in facebook. In *Proceedings of the 2nd ACM workshop on Online social networks*, pp. 37–42, 2009.

Dennis Wagner, Arjun Nair, Billy Joe Franks, Justus Arweiler, Aparna Muraleedharan, Indra Jungjohann, Fabian Hartung, Andriy Balinskyy, Saurabh Varshneya, Mayank Chetan Ahuja, Nabeel Hussain Syed, Mayank Nagda, Philipp Liznerski, Steffen Reithermann, Maja Rudolph, Sebastian Josef Vollmer, Ralf

Schulz, Torsten Katz, Stephan Mandt, Michael Bortz, Heike Leitte, Daniel Neider, Jakob Burger, Fabian Jirasek, Hans Hasse, Sophie Fellenz, and Marius Kloft. Formally exploring time-series anomaly detection evaluation metrics. In *The 29th International Conference on Artificial Intelligence and Statistics*, 2026. URL https://openreview.net/forum?id=INJj1SB5Uw.

Xiaoqi Wang and Han Wei Shen. GNNInterpreter: A probabilistic generative model-level explanation for graph neural networks. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=rqq6Dh8t4d.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, pp. 1907–1913. AAAI Press, 2019. ISBN 9780999241141.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32, 2019.

Qi Zhang, Jianlong Chang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Spatio-temporal graph structure learning for traffic forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 1177–1185, 2020.

A Tensor-based dynamic mode decomposition

As mentioned in Section 2.1.1, the standard DMD algorithm relies on the vectorisation of the snapshots $\boldsymbol{h}_t$ to build the matrices in (4), incurring the curse of dimensionality.

Instead of relying on dimensionality reduction techniques like PCA to mitigate this issue, an alternative is to use a more efficient algebraic representation that leaves the original dimensions intact. One such method is TT-DMD (Klus et al., 2018), which overcomes high computational costs by exploiting the tensor-train format (Oseledets, 2011) for the snapshot tensors (4). Importantly, TT-DMD does not perform dimensionality reduction; rather, it factorises the data to make full-dimensional computations tractable.

In the tensor-train representation, a tensor $\boldsymbol{X}$ of order d is decomposed into a tensor product of d tensors of order at most 3, called *cores*:

$$\boldsymbol{X} = \sum_{k_0=1}^{r_0} \cdots \sum_{k_d=1}^{r_d} \boldsymbol{X}^{(1)}_{k_0,:,k_1} \otimes \boldsymbol{X}^{(2)}_{k_1,:,k_2} \otimes \cdots \otimes \boldsymbol{X}^{(d-1)}_{k_{d-2},:,k_{d-1}} \otimes \boldsymbol{X}^{(d)}_{k_{d-1},i_d,k_d}, \tag{27}$$

or, focusing on the entries,

$$\boldsymbol{X}_{i_1,\ldots,i_d} = \sum_{k_0=1}^{r_0} \cdots \sum_{k_d=1}^{r_d} \boldsymbol{X}^{(1)}_{k_0,i_1,k_1} \cdot \boldsymbol{X}^{(2)}_{k_1,i_2,k_2} \cdot \cdots \cdot \boldsymbol{X}^{(d-1)}_{k_{d-2},i_{d-1},k_{d-1}} \cdot \boldsymbol{X}^{(d)}_{k_{d-1},i_d,k_d}. \tag{28}$$

The advantage of this representation is twofold: on one hand, it is possible to rewrite the DMD algorithm so that it takes advantage of the tensor-train format. On the other hand, each dimension of the original tensor is stored on a different core and therefore retains its meaning, unlike the standard vectorisation approach, which mixes dimensions.

This approach is related to the *tensor PCA* introduced in (Bianchi et al., 2021) in the context of reservoir computing, where dimensionality reduction is applied to the feature mode of the reservoir states tensor while preserving the temporal structure.

In our setting, we transform the STGNN's embeddings $\boldsymbol{h}_t \in \mathbb{R}^{N \times F}$ into the tensor-train format, and then we apply TT-DMD.

B Description of datasets

The semi-synthetic datasets employed in the experiments consist of TGs whose time-varying topologies describe different types of social interactions. The *Facebook* dataset is based on the activity of the New Orleans Facebook community over three months (Viswanath et al., 2009). The *Infectious* dataset is based on face-to-face contacts between visitors of the SocioPattern project (Isella et al., 2011). The *DBLP* dataset is based on co-author graphs from the DBLP database, with publication year used as the timestamp. The *Highschool* dataset is based on a contact network from the SocioPattern project, describing interactions between high school students over seven days. The *Tumblr* dataset is based on a graph of quoting interactions between Tumblr users (Leskovec et al., 2009).

Table 4: Description of datasets.

Dataset	$N_{\mathcal{G}}$	T	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	C
Facebook	995	106	71–100	176–362	2
Infectious	200	50	50	218–1010	2
DBLP	755	48	50–60	96–380	2
Highschool	180	205	26–60	302–1178	2
Tumblr	373	91	25–99	96–380	2
MSRC-12	6243	14–493	20	58	12

The original MSRC-12 dataset consists of 594 sequences in which 30 people perform 12 actions, captured at a 30 Hz sampling rate. Since in each sequence the action is performed multiple times, we preprocess the dataset so that each input sequence corresponds to a single action.

Table 4 reports the details of each dataset, such as the number of TGs $N_{\mathcal{G}}$, the range of the length of the temporal sequences T, the minimum and maximum number of nodes $|\mathcal{V}|$ and edges $|\mathcal{E}|$, and the number of classes C.

C Hyperparameters and implementation details

The presented methods depend on several hyperparameters. Some of these are used to define the architecture and the training of the STGNN, others are involved in the explainability methods. Table 5 shows all possible values, and Tables 6 to 8 report the optimal hyperparameter configurations used for each dataset. In order to find the best values, we perform a grid search. For those parameters related to the model's architecture and training, we select the values that yield the highest performance in terms of classification accuracy. For the parameters related to the explainability methods, we use the F1 score defined in Section 4.2 as the validation metric.

D Saliency baselines

To strengthen the empirical comparison, we compare our results with explanations produced by a saliency map (Simonyan et al., 2014). We use a standard saliency map to find the nodes and times of the input TG that are most relevant to the model. The saliency map provides a saliency attribute

$$S(n,t) := \left|\nabla_{\boldsymbol{x}_{t,n}} y\right|, \tag{29}$$

where y is the output of the model (14), and $\boldsymbol{x}_{t,n}$ is the input label of the n-th node. Notice that we only consider node labels, so we don't compute a saliency attribute for the input's edges. We can define a temporal explanation as $w_{\mathtt{t}}(t) = \sum_n S(n,t)$ and, as done before, we can measure the F1 score by highlighting those time steps t such that $w_{\mathtt{t}}(t) > \delta$, where we use median absolute distance (MAD) as threshold. The use of MAD is necessary due to the shape of $w_{\mathtt{t}}(t)$, which shows some peaks much more prominent than others, and the other methods would hinder weaker, but relevant, peaks.

The spatial explanation is defined as

$$w_{\mathtt{s}}(n) := \max_t |S(n,t)|. \tag{30}$$

The measured values of the F1 scores and AUC are reported in Tables 2 and 3, referred to as F1sal and AUC_{sal}, and they offer a baseline for the metrics F1 and $\text{AUC}_{\mathcal{G}}$ respectively.

E Further examples from MSRC-12

We report in this section, in Figures 8 to 16, one example of an explanation on MSRC-12 for each class not discussed in Section 4.3. Here are some comments:

- Most of the classes in the dataset involve movements of the upper body, especially the arms (e.g. "raise volume of music", "put on goggles", "wind up the music", etc.). For this reason, the explanations are similar, but some details serve as telltale signs of how the model can differentiate between them. For example, in Figure 8, more weight is given to the hands than in Figures 10 and 11.
- It is interesting to compare the classes "navigate to next menu" in Figure 9 and "throw an object" in Figure 13. Both primarily involve the movement of one arm, but in both cases, surprisingly, the model focuses mainly on the arm that remains still: in the first case, the hand nodes; in the latter, the whole arm and the pelvis.

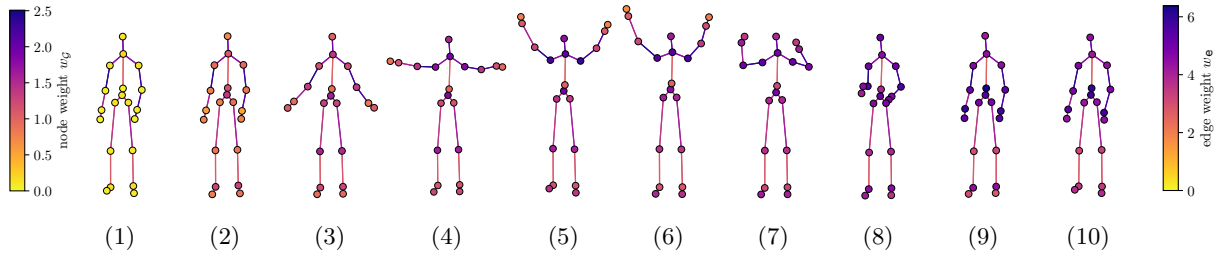


Figure 8: One example from the MSRC-12 dataset, corresponding to the class "raise volume of music".

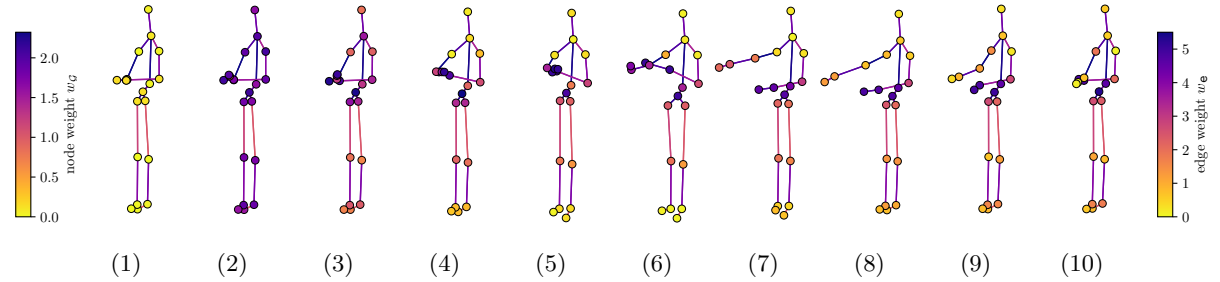


Figure 9: One example from the MSRC-12 dataset, corresponding to the class "navigate to next menu".

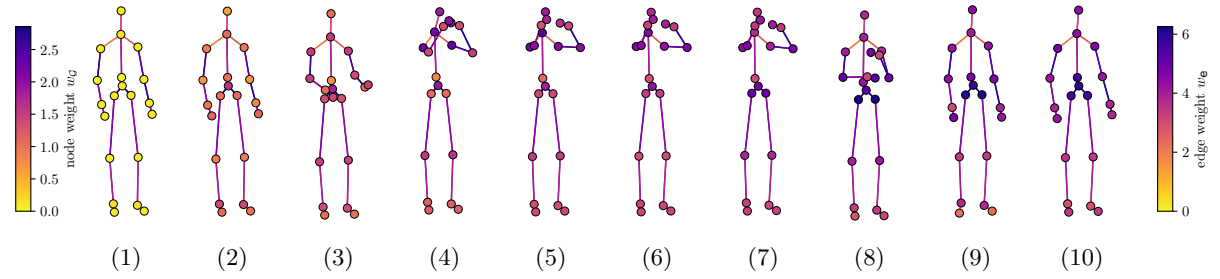


Figure 10: One example from the MSRC-12 dataset, corresponding to the class "put on goggles".

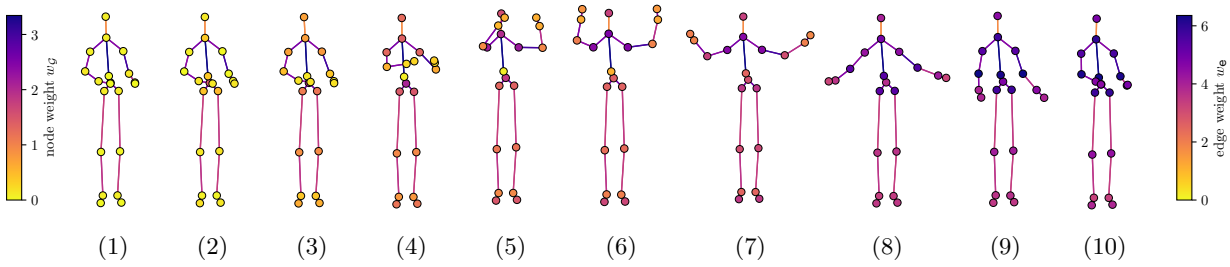


Figure 11: One example from the MSRC-12 dataset, corresponding to the class "wind up the music".

- The "kick" class in Figure 16 has a more distinct motion, making the explanation also more understandable: it highlights the spine and the leg involved in the kicking motion.

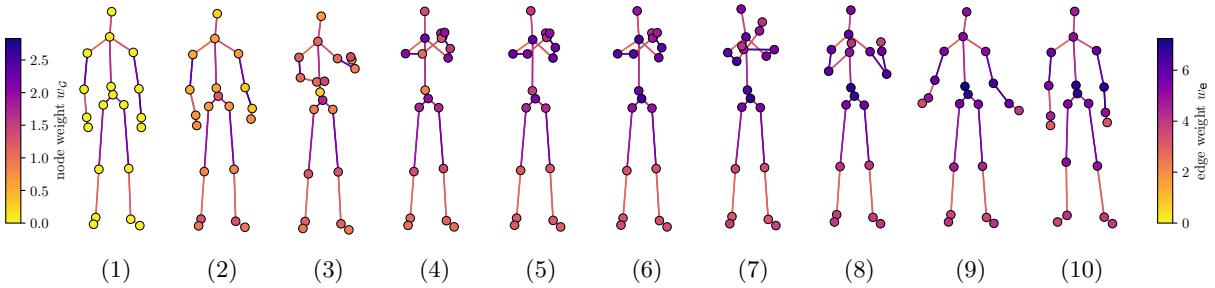


Figure 12: One example from the MSRC-12 dataset, corresponding to the class "shoot a pistol".

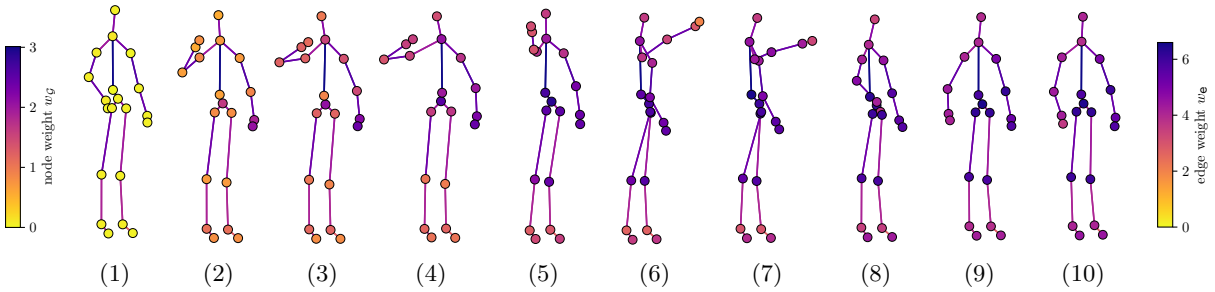


Figure 13: One example from the MSRC-12 dataset, corresponding to the class "throw an object".

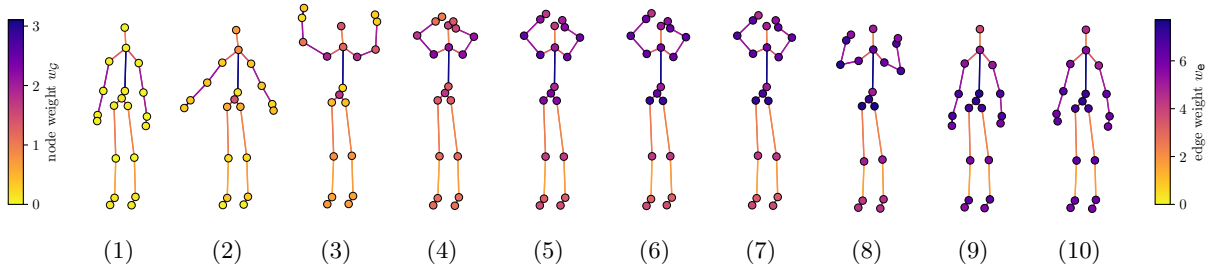


Figure 14: One example from the MSRC-12 dataset, corresponding to the class "protest the music".

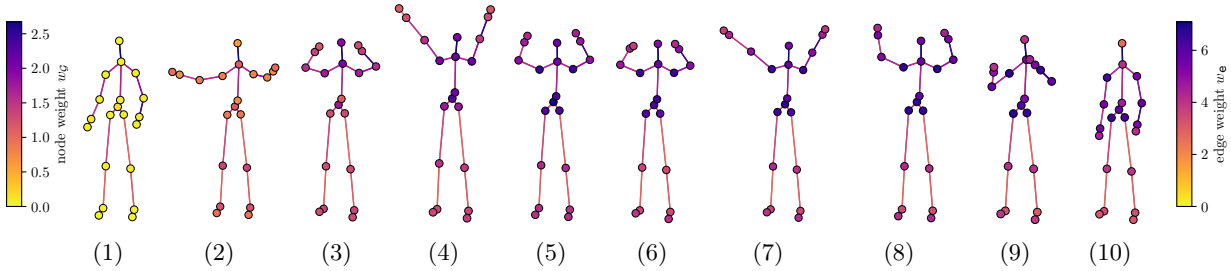


Figure 15: One example from the MSRC-12 dataset, corresponding to the class "move up the tempo of the song".

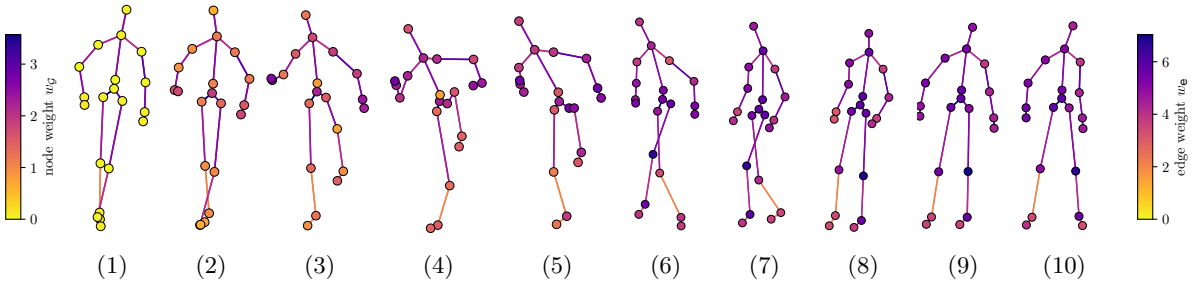


Figure 16: One example from the MSRC-12 dataset, corresponding to the class "kick".

F Regularised STGNN with linear dynamics

In general, the sequence of states $\boldsymbol{h}_{t,n}$ does not evolve according to the linear dynamics of (2). To investigate this, we introduce an internal state

$$\tilde{\boldsymbol{h}}_{t+\tau,n} := \boldsymbol{K}^{\tau} \boldsymbol{h}_{t,n}, \tag{31}$$

where $\boldsymbol{K} \in \mathbb{R}^{F \times F}$ is a trainable parameter that acts as Koopman operator on $\tilde{\boldsymbol{h}}_{t,n}$. Moreover, to train $\boldsymbol{K}$, we add the following terms to the loss:

- a binary cross-entropy term ℓ_{rec} between the Koopman-reconstructed output $\tilde{y} = \text{MLP}(\tilde{\boldsymbol{h}})$ and the class label $\hat{y}$;
- an observable loss ℓ_{obs}, defined as a mean-squared reconstruction loss between the TG embedding at time t, $\boldsymbol{h}_t$, and the corresponding Koopman-reconstructed embedding $\tilde{\boldsymbol{h}}_t$, together with an ℓ_2 penalty on $\boldsymbol{K}$:

  $$\ell_{\text{obs}}(\boldsymbol{h}_t, \tilde{\boldsymbol{h}}_t) = \text{MSE}(\boldsymbol{h}_t, \tilde{\boldsymbol{h}}_t) + \ell_2(\boldsymbol{K}), \tag{32}$$

  where ℓ_2 is a weight decay regularisation term.

These two terms represent regularisation losses that push the model to represent an observable φ that satisfies the Koopman operator definition (2), as proposed by (Li et al., 2017) and (Lusch et al., 2018) in a deep learning setting.

We note that the sole purpose of $\boldsymbol{K}$ and the internal state $\tilde{\boldsymbol{h}}_{t,n}$ is to encourage the state $\boldsymbol{h}_{t,n}$ to follow $\tilde{\boldsymbol{h}}_{t,n}$, whose dynamics is, by construction, linear. They are not used to produce the output y, nor are they involved in explaining the model.

The complete loss then becomes

$$\ell = \ell_{\text{ce}} + \alpha \ell_{\text{rec}} + \beta \ell_{\text{obs}}, \tag{33}$$

where α and β are hyperparameters.

The proposed model is depicted in Figure 17.

We aim to test whether the proposed regularisation in (33), which pushes the model dynamics

$$\boldsymbol{h}_{t+1} = \text{STGNN}(\boldsymbol{h}_t, \boldsymbol{x}_t, \boldsymbol{A}_t) \tag{34}$$

to exhibit an approximately linear behaviour, improves the performance of the proposed explainability methods. To do so, we perform an ablation study on the parameters α and β.

In Figures 18 to 21, we present results obtained by training a GCRN model on four datasets[5], varying the values of α and β among 0, 0.1, 0.5, 1, and 5, over five different seeds. Overall, the regularisation terms have little and inconsistent effects:

[5]The Highschool dataset is omitted from this ablation solely because training a model on it requires substantially more time.

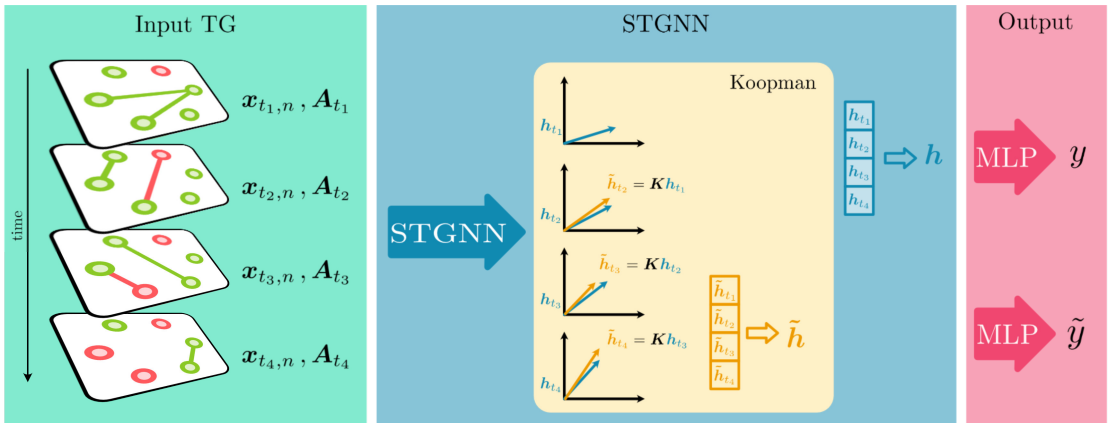


Figure 17: The left-hand side, in green, depicts an example of a TG, with node features $\boldsymbol{x}_t$ and adjacency matrix $\boldsymbol{A}_t$. In blue, the STGNN processes the input and provides an embedding $\boldsymbol{h}_t$ for each time step. The inner yellow box represents the mechanism that encourages the embeddings dynamics $\boldsymbol{h}_t$ to be linear: the loss ℓ_{obs} in (32) pushes $\boldsymbol{h}_{t+1}$ to be a linear transformation of $\boldsymbol{h}_t$ (for illustration, the figure shows a 2-dimensional rotation). In red, an MLP produces the final output.

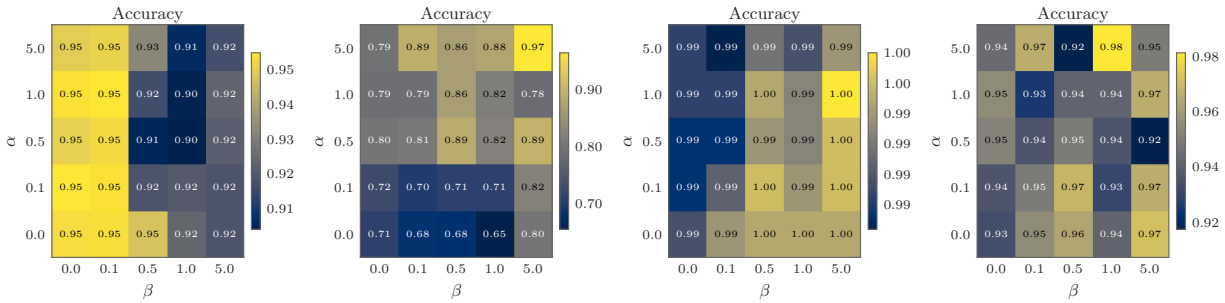
Accuracy In Figure 18, there are some noticeable positive effects on the Infectious and Facebook datasets, while in the other cases, the variations are small or don't follow a clear pattern.

F1 In Figure 19, unlike accuracy, a pattern is visible for DBLP and Tumblr, albeit still with small variations.

AUC$_\mathcal{G}$ In Figure 20, there are some small improvements in DBLP and Tumblr, while the effect of the regularisation is null or detrimental in Facebook and Infectious.

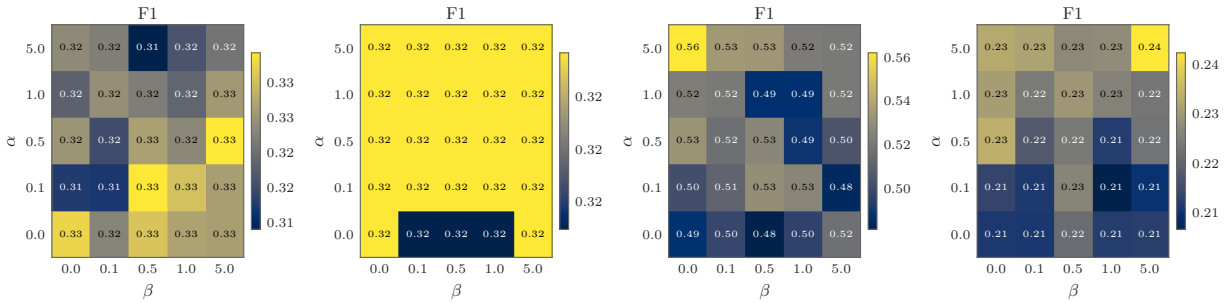
AUC$_{\text{edge}}$ In Figure 21, as for F1 and AUC$_\mathcal{G}$, there are some small improvements in DBLP and Tumblr, but zero to negative effects for Facebook and Infectious.

These results provide two interesting insights. On the one hand, they indicate that the explainability gains reported in the main text do not depend on explicitly enforcing linear latent dynamics during training: the proposed post hoc analysis can already extract meaningful structure from standard STGNNs. On the other hand, they suggest that the relationship between more structured latent dynamics and better explanations is more subtle than can be captured by a simple auxiliary loss. In this sense, this ablation helps delimit the contribution of the paper and motivates future work on more targeted ways of introducing dynamical priors into temporal graph models. For this reason, we keep the simpler formulation in the main text and report this variant here as an exploratory extension.



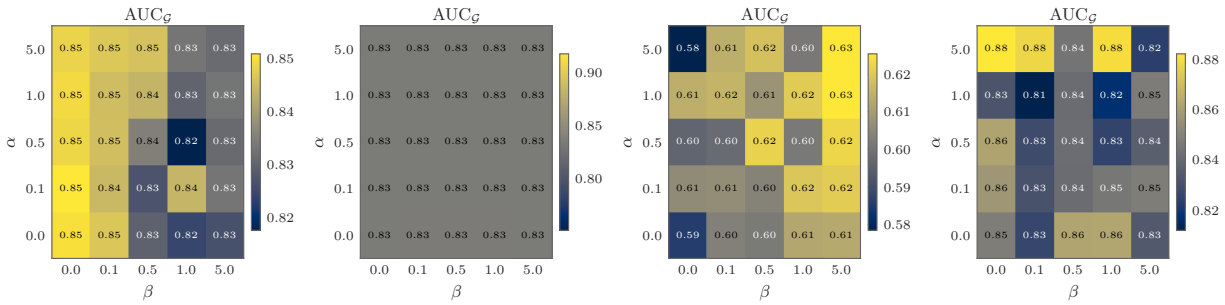
(1) Facebook (2) Infectious (3) DBLP (4) Tumblr

Figure 18: Effect of α and β on accuracy.



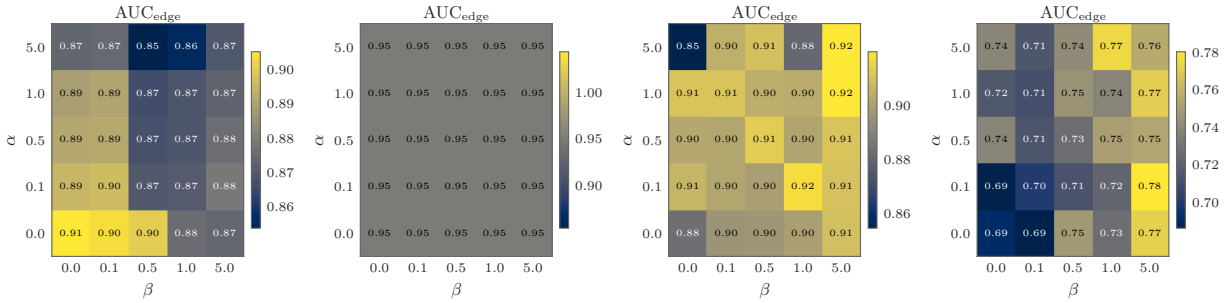
(1) Facebook (2) Infectious (3) DBLP (4) Tumblr

Figure 19: Effect of α and β on F1.



(1) Facebook (2) Infectious (3) DBLP (4) Tumblr

Figure 20: Effect of α and β on AUC$_{\mathcal{G}}$.



(1) Facebook (2) Infectious (3) DBLP (4) Tumblr

Figure 21: Effect of α and β on AUC$_{\text{edge}}$.

Table 5: Candidate values for hyperparameters.

	Parameter	Candidates
STGN par.	RNN type	LSTM, GRU
	Activation	identity, linear, ReLU, leaky ReLU, tanh
	F	16, 32, 64
	L	$1, \ldots, 10$
	K (DCRNN, GWN)	$2, 3, 4, 5$
	$K_\mathtt{t}$ (GWN)	$2, 3, 4, 5$
	Dilation d (GWN)	$1, 2, 4$
	MLP layers	$1, \ldots, 5$
Expl. par.	Dim. red.	PCA, SVD, TT-DMD
	f	10, 16, 32, 64
	F1 type	threshold (THR), window (WIN)
	Threshold type	MAX, AVG or MAD
	ω	$2, \ldots, 6$
	d_{SINDy}	$2, 3$
	Mode i	$0, 1$

Table 6: Selected hyperparameters for GCRN.

	Parameter	Facebook	Infectious	DBLP	Highschool	Tumblr	MSRC-12
STGNN par.	RNN type	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM
	Activation	linear	linear	ReLU	ReLU	linear	identity
	F	64	64	64	32	64	32
	L	2	2	4	3	3	3
	MLP layers	1	1	3	4	1	1
Expl. par.	Dim. red.	PCA	TT-DMD	TT-DMD	TT-DMD	PCA	TT-DMD
	f	10	10	10	10	10	10
	F1 type	THR	THR	THR	THR	WIN	—
	Thr. type	AVG	AVG	MAD	MAD	AVG	—
	ω	5	5	5	5	5	5
	d_{SINDy}	2	3	2	3	3	3
	Mode i	0	0	0	0	0	0

Table 7: Selected hyperparameters for DCRNN.

	Parameter	Facebook	Infectious	DBLP	Highschool	Tumblr	MSRC-12
STGNN par.	RNN type	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM
	Activation	ReLU	tanh	ReLU	linear	linear	leaky ReLU
	F	32	32	64	16	32	64
	K	2	2	2	2	5	5
	L	1	1	1	1	1	2
	MLP layers	1	3	2	2	3	1
Expl. par.	Dim. red.	PCA	TT-DMD	TT-DMD	TT-DMD	PCA	TT-DMD
	f	10	10	10	10	10	10
	F1 type	THR	THR	THR	THR	WIN	—
	Thr. type	AVG	AVG	MAD	MAD	AVG	—
	ω	5	5	5	5	5	5
	d_{SINDy}	2	3	3	3	3	3
	Mode i	0	0	0	0	0	0

Table 8: Selected hyperparameters for GWN.

	Parameter	Facebook	Infectious	DBLP	Highschool	Tumblr	MSRC-12
STGNN par.	Activation	leaky ReLU	linear	leaky ReLU	linear	leaky ReLU	ReLU
	F	16	32	64	64	64	32
	L	8	5	4	4	5	5
	K	3	4	4	2	4	3
	$K_\mathtt{t}$	5	3	2	5	5	5
	d	2	4	1	4	4	4
	MLP layers	3	3	3	3	2	1
Expl. par.	Dim. red.	PCA	PCA	PCA	TT-DMD	PCA	TT-DMD
	f	10	10	10	10	10	10
	F1 type	WIN	THR	THR	WIN	WIN	—
	Thr. type	AVG	MAD	MAD	AVG	AVG	—
	ω	5	5	5	5	5	5
	d_{SINDy}	3	3	3	3	3	3
	Mode i	0	0	0	0	0	0