Applied Network Science

RESEARCH Open Access

A computational framework for modeling complex sensor network data using graph signal processing and graph neural networks in structural health monitoring

Stefan Bloemheuvel[1,2], Jurgen van den Hoogen[1,2] and Martin Atzmueller[3*]

*Correspondence: martin.atzmueller@uni-osnabrueck.de
[3] Semantic Information Systems Group, Osnabrück University, Osnabrück, Germany
Full list of author information is available at the end of the article

Abstract

Complex networks lend themselves for the modeling of multidimensional data, such as relational and/or temporal data. In particular, when such complex data and their inherent relationships need to be formalized, complex network modeling and its resulting graph representations enable a wide range of powerful options. In this paper, we target this—connected to specific machine learning approaches on graphs for *Structural Health Monitoring* (SHM) from an analysis and predictive (maintenance) perspective. Specifically, we present a framework based on *Complex Network Modeling*, integrating *Graph Signal Processing* (GSP) and *Graph Neural Network* (GNN) approaches. We demonstrate this framework in our targeted application domain of SHM. In particular, we focus on a prominent real-world SHM use case, i. e., modeling and analyzing sensor data (strain, vibration) of a large bridge in the Netherlands. In our experiments, we show that GSP enables the identification of the most important sensors, for which we investigate a set of search and optimization approaches. Furthermore, GSP enables the detection of specific graph signal patterns (i. e., mode shapes), capturing physical functional properties of the sensors in the applied complex network. In addition, we show the efficacy of applying GNNs for strain prediction utilizing this kind of sensor data.

Keywords: Complex networks, Graph signal processing, Sensor data, Complex networks for physical infrastructures, Structural health monitoring, Graph neural networks, Machine learning on graphs

Introduction

For modeling complex data, e. g., continuous sequential, multi-relational and heterogeneous data, graphs provide sophisticated means for modeling and representation. In particular, for modeling complex systems—including those providing complex sensor data—graphs have emerged as a natural representation. Here, *Graph Signal Processing (GSP)* (Stankovic et al. 2019a) has recently emerged as a powerful analytical framework in such contexts: this is enabled both at the level of the network structure, as well as its (temporal) dynamics; GSP specifically extends on classical signal processing by

providing specific analytical options on irregular structures such as graphs and networks (Shuman et al. 2013), which naturally accounts for irregular relations (Stankovic et al. 2019b). Besides GSP, Deep Learning approaches have been adopted in complex network modeling and analysis as well. Here, we specifically apply a Graph Convolutional Network (Kipf and Welling 2017; Wu et al. 2020) (GCN) approach, where one of its branches of origin is actually rooted in GSP (Cheung et al. 2020), the so-called spectral-based Graph Neural Networks (GNNs).

Overall, in this paper—a substantially adapted and extended revision of Bloemheuvel et al. (2020)—we present a computational framework for modeling complex sensor data in the form of complex networks including GSP and GNN for Structural Health Monitoring (SHM) (Miao 2014; Sony et al. 2019; Abdulkarem et al. 2020) and analysis. Compared to Bloemheuvel et al. (2020), we have specifically extended the presentation of the proposed approach, the contextualization as well as the experimentation. Most importantly, we have included a novel component into our framework, i. e., the GNN-based method for incorporating predictive analytics into our computational framework.

SHM is a multi-disciplinary field applying data-driven diagnostic methods which aim at investigating and estimating the integrity of massive complex structures. For these, it is then the ultimate goal to increase safety, reliability, efficiency, and ultimately (cost-) effectiveness in such contexts, e. g., relating to civil infrastructures such as pipeline systems, buildings, and bridges. SHM data typically includes discrete-domain signals (time series). Adapting and using insights and methods from civil engineering, signal processing, sensor technology, machine learning and data mining, cf. Miao (2014), the data can then be analyzed. To the best of the authors' knowledge, as presented in this paper, this is the first time that a combination of GSP and GNNs has been applied for such a data modeling and analysis task with respect to complex networks on real-world physical infrastructures.

Utilizing our proposed computational framework, we apply GSP and GNNs for SHM using a real-world dataset which has been collected in the context of a SHM project in the Netherlands called *InfraWatch* (Knobbe et al. 2010). In this project, data has been captured by a set of sensors installed on a major highway bridge (the so-called *Hollandse Brug*), estimating the properties of traffic (i. e., pressure) which is passing over the bridge. For this, sensors estimating strain, vibration, as well as ultrasonic wave sensors (Lynch and Loh 2006) are typically directly attached to the respective structure.

There are two main advantages when analyzing and optimizing sensor networks (Capellari et al. 2018). To begin with, by optimizing the sensor network with respect to sensor location and type, the number of sensors can be reduced by sampling the most optimal subset. This leads to a cost reduction in the total SHM system. Furthermore, the amount of data that has to be analyzed is reduced significantly, speeding up the analysis. Besides that, it also increases the possibility to create real-time estimation models and reduces the data storage in the long term (Capellari et al. 2018). Furthermore, GSP allows for the detection of unique trends in complex data as well as the recognition of specific events. This includes, e. g., the identification of traffic peaks and complex patterns found when a significant amount of pressure is applied to different sections of the bridge. Such patterns can then indicate implicit/explicit hints and information for assessing the health of the bridge (Seo et al. 2016).

Both of these problems are addressed in this work, i. e., how modeling and analysis are carried out and to what degree we can identify certain subsets of sensors as well as interesting patterns in the modeled complex data, respectively. In addition, GNN methods can assist with the use case of real-time condition monitoring by forecasting the localized structural strain response of a respective monitored object. This structural strain response has recently received increased attention in the context of Condition Health Monitoring and prognosis since continuous strain measurements can then provide insights about the stress experienced on the bridge, in order to better characterize local weaknesses and damage to the structure compared to global responses (Wan and Ni 2018). This highlights the potential of accurate forecasting of the structural stress responses by GNNs that (1) incorporate modern Deep Learning techniques and (2) complex networks to incorporate the spatial interdependence between the sensors.

Our contributions are outlined below:

1. We suggest a theoretical framework for SHM that includes network modeling as well as complex network analysis using GSP and GNN approaches.
2. We outline the complex network modeling and analytical methodology of the presented framework in detail, exemplified by our application use case.
3. We use a dataset comprising real-world sensor data modeled in a complex network to illustrate the implementation of this system in a case study.

    (a) We present comprehensive analysis results for sensor network modeling in a resource-aware manner, with the aim of using the fewest number of sensors possible to recreate the provided signals.
    (b) We present modeling results for signal pattern and event recognition.
    (c) We show the ability of GNNs to grasp the physical nature of the sensors on a complex network embedded on a bridge.

The remainder of this work is organised as follows: Section 2 addresses related work, including an outline of essential theoretical concepts of GSP and GNN theory. Section 3 then introduces our proposed structure and explains the approach in depth. Section 4 introduces the case study and addresses our results. Finally, Section 5 concludes with an overview and interesting directions for future research.

Background and related work

This section discusses related work and outlines important fundamental concepts on the background of our proposed framework. We start by summarizing related work on complex networks, before we introduce the fundamental concepts of signal processing on graphs and the requisite theoretical context. For a detailed overview on GSP, we refer to e. g., Ortega et al. (2018), Stankovic et al. (2019a). Next, we focus on the topic of Structural Health Monitoring. Finally, we provide a brief summary on Graph Neural Networks, where we introduce and explain this prominent approach for Deep Learning on graphs.

Complex networks

In the world of today, complex networks—represented as graphs—can be observed in many different areas and domains. Altogether, complex networks have proven to be an effective method for modeling structural properties in a wide range of complex systems and a number of domains (e. g., Strogatz 2001; Amaral and Ottino 2004; Boccaletti et al. 2006; Mitzlaff et al. 2014; Atzmueller 2014; Bloemheuvel et al. 2019). In particular, complex structures encountered in complex (networked) systems and structures, such as computer networks, social networks, infrastructure networks, sensor networks, as well as cyber-physical networks play an important role throughout our everyday life. However, the network concept transcends such explicit structures, towards more implicit networks observed in physical structures of interdependent elements or components (Bloemheuvel et al. 2020; Worden 2021). In particular, in the field of complex networks and feature rich networks both the need as well as the opportunities in studying such complex network topologies, has made the use of complex network models pervasive in many fields of research such as computer science, physics, engineering and the social sciences, also joining into interdisciplinary research contexts (cf. Interdonato et al. 2019).

In comparison to simple homogeneous static networks, real-world networks are often dynamic and heterogeneous, with both nodes and links being represented by a collection of attributes and/or complex relationships caused by multi-relational, continuous sequential, and heterogeneous data. Thus, the mining of so-called *feature-rich* networks (Interdonato et al. 2019) is gaining increasing interest; such networks include, in particular, *node-attributed* and/or *edge-attributed* networks, where, for example, time series information obtained from sensor readings can be attached to nodes and/or edges of a network.

In this paper, we target the modeling of complex sensor network data—regarding topological/structural dependencies and properties using complex network approaches. Specifically, we apply GSP and GNNs on the modeled networks (being represented as graphs). To the best of the authors' knowledge, this is the first time that such a combination of modeling and analysis methods has been applied for the task of SHM on real-world physical infrastructures.

Graph signal processing

Classical signal processing can be exceptionally strong in uniform, euclidean domains, e. g., in the context of audio and power circuits. However, not all domains possess such a desirable feature. For instance, if examining sensors arranged along some topography of a building at distinct locations, then this arrangement will in all likelihood not resemble some kind of regular grid, where, e. g., wall and floor properties can considerably influence positioning and signal strengths of sensors. Moreover, transportation networks also resemble complex connections that are not structured uniformly. Some locations will serve as hubs in the network of rails, while there will be less dense connections in more urban areas. Thus, the complexity of such networks implies that the data coming from such irregular and complex structures do not lend themselves for standard tools (Ortega et al. 2018). This motivates, e. g., including the

spatial dimension towards complex modeling via GSP, extending signal processing by including irregular structures modeled as graphs (Shuman et al. 2013). Signal data on a graph can then be intuitively represented as a finite set of samples, where each node contained in the graph is assigned to one sample.

GSP: Basic Definitions We define a graph as $G = (V, E)$ where V are the nodes (also called vertices) and E the edges. An edge $e_{ij} = (v_i, v_j)$ connects nodes v_i and v_j, i. e., they are neighbors. The adjacency matrix $A \in \mathbb{R}^{N \times N}$ where $|V| = N$ is a square matrix such $A_{ij} = 1$ if there is an edge from node v_i to node v_j, and 0 otherwise. The number of neighbors of a node v is known as the degree of v and is denoted by $D_{ii} = \sum_j A_{ij}$. For GSP, a graph G is most often represented via the Laplacian matrix $L \in \mathbb{R}^{N \times N}$, i. e., the degree matrix minus the adjacency matrix; it holds several spectral properties that are desirable during GSP analysis (Stankovic et al. 2019a). For example, the Laplacian of an undirected graph is always positive semi-definite (all the eigenvalues of the matrix are non-negative). For a more detailed overview (see Stankovic et al. 2019a; Ortega et al. 2018; Ruiz et al. 2021).

- A graph signal is defined by associating real data values s_n to each vertex. A graph signal is written as $s = [s_0, s_1, \ldots, s_{N-1}]^{\mathrm{T}} \in \mathbb{R}$ in vector notation.
- In Digital Signal Processing, a signal shift is a shift in time of length N, resulting in $\hat{s} = s_{n-1}$. Such an operation helps with performing autocorrelation analysis. In GSP, a signal shift is more locally defined by replacing a signal value by a combination of a neighbors signal values V_n weighted by their respective edge weights. The two most popular graph shift operators are given by the Laplacian and adjacency matrix.
- One of the most important transformations in classical Signal Processing is the Fourier transform, which changes the domain of a signal x from the time-domain to the frequency-domain. This change of perspective makes previously difficult problems more easily solvable, since it tells you what frequencies are present in your signal and in what proportions. Translated in terms of GSP, the Graph Fourier Transform (GFT) converts the graph signal from the vertex domain into the graph spectral domain. GSP achieves this transformation via the spectral decomposition of

$$L = V \Lambda V^{-1}, \tag{1}$$

  where the columns v_n of the matrix V are the eigenvectors of the Laplacian L, and Λ the diagonal matrix of the corresponding eigenvalues. The eigenvalues act as the frequencies on the graph (Sandryhaila and Moura 2014). The GFT of the signal s is then calculated by $\hat{s} = U^* s$ where U^* the conjugate transpose of the Fourier Basis U.
- After Graph Fourier transformation, filters can be applied. These filters transform the graph signal into the graph spectral domain. Then, unwanted frequencies are weakened or wanted frequencies are magnified by altering the Fourier coefficients. Finally, the signal is reverted to the vertex domain.
- Lastly, a technique to measure the smoothness of a signal on a graph is called Total Variation. Smoothness is an important subject in Graph Signal Processing since a lot of techniques depend on the assumption that nearby nodes act similar. Smoothness is expressed by the function:

$$\mathrm{TV}_{\mathrm{G}}(\mathbf{X}) = \|\mathbf{X} - \mathbf{A}\mathbf{X}\|_1 \tag{2}$$

where A is the shift operator matrix of the graph, AX the shifted version of the signal and $\|\|_1$ the l_1-norm. In other words, it is the cumulative difference between the original signal at each node and its neighbors. One could then use the end result as a global measure for the entire signal, or also investigate the individual values for each sensor.

Structural health monitoring

The collapse of the Polcevera Viaduct in Genoa showed that good designs alone are insufficient to ensure long-term viability of civil infrastructure (Clemente 2020). Such structures should be continually checked to identify damage and defects and to schedule timely maintenance programs. The field of applying such data-driven diagnostics that investigate and estimate the integrity of massive structures is called Structural Health Monitoring (SHM).

In principle, the main assumption of SHM is that global parameters (e. g., mode shapes, natural frequencies) are functions of physical properties such as mass, damping, and stiffness (Seo et al. 2016; Cornwell et al. 1999). The deformation that a part will exhibit when vibrating at its natural frequency is referred to as its mode shape. From a Signal Processing view, mode shapes are patterns where signals and their frequencies are partitioned into different modal categories, e. g., using strain and/or vibration sensors. Both local and global characteristics can be extracted. Specific local abnormalities of sensor data, for example, can suggest inaccurate sensor readings, motivating sensor replacement and/or maintenance. Global characteristics, however, could assess changes in the overall stiffness of a structure (Seo et al. 2016), and determine the current and future structural capacity of a bridge (Seo et al. 2016).

A specific problem connecting GSP and SHM which we also target in this paper, is resource-aware optimization; via identifying the minimal subset of sensors which is required to reconstruct the signal using GSP (Capellari et al. 2018), the needed number of sensors for reliably capturing (sensor) data from a specific complex system (e. g., a bridge) can be minimized, e. g., for a sensor network monitoring dynamic/structural properties.

Graph neural networks

Besides GSP, Graph Neural Networks (GNNs) have emerged as another successful technique for modeling complex graph-structured data. One of the branches of origins in GNNs called spectral-based GNNs even originates from the GSP literature. Older efforts to build GNNs mostly consist of spatial methods that look at the neighborhood of nodes to perform message passing between pairs of nodes to agglomerate them. This gap (spatial/spectral) has been bridged by the Graph Convolutional Network (GCN) (Kipf and Welling 2017). Since then, spatial-based techniques have developed rapidly due to their efficiency and generality (Wu et al. 2020), e. g., Graph Convolutional Networks, Graph Autoencoders and Spatial-Temporal Graph Neural Networks; here, Graph Convolutional Networks gained most attention (Wu et al. 2020).

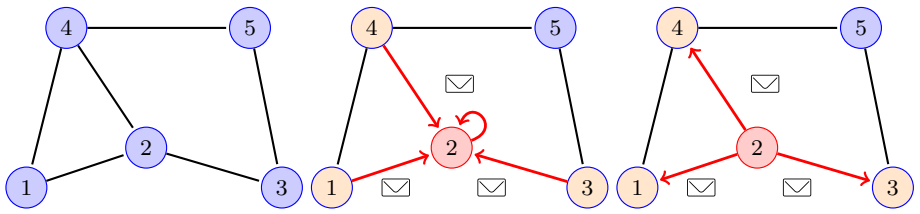


Fig. 1 Example of the basic message passing procedure of a GNN/GCN

Essentially, almost all the GNNs can be expressed as Message Passing Neural Networks (Gilmer et al. 2017).

1. The message passing function defines how the convolution works;
2. a node update function determines the new node states after propagation;
3. a readout function determines what is done with this information (e.g., node classification or link prediction).

Figure 1 depicts a simple schematic overview of the node updating procedure, and its respective steps. First, node 2 will collect the node feature information from its neighbors. Then, it will update its state and also provide a message for its own neighbors, concluding the proposition of Gilmer et al. (2017).

The spectral-based GNNs exploit the adjacency or Laplacian matrix and the degree matrix of a graph to perform the convolution in the Fourier domain, similar to GSP techniques. A graph signal is convoluted throughout the graph in the Fourier domain, and reversely transformed back to the graph domain. However, a severe limitation of spectral-based methods is the lack of Transferability, since the method is dependent on the specific graph it is trained on. Therefore, the graph neural network models also needs the entire graph to train on, which is more complex in larger graph settings.

Spatial-based methods define the graph convolution on the spatial relations of a node, similar to the convolution step in a conventional CNN with image data. The graph convolution combines the representation of the central node's representation with its neighbors representations to derive the updated state of the central node. The spatial graph convolutional operation fundamentally propagates node information along edges. Below, we summarize the core mechanisms which are relevant for the methods applied in this paper.

In both types of convolutions, added to this propagation of information are optional node and edge features. These node features and edge features in a graph $G = (V, E)$ are the feature description x_i for every node i in the $V \times F$ matrix X, where F is the number of input features.

However, to work with an adjacency matrix and to use node features and edge features, some adaptions have to be made to the classical way a neural network performs feature propagation. In normal neural networks, we propagate to the next layer by:

$$H^{i+1} = \sigma\left(W^i H^i + b^i\right), \tag{3}$$

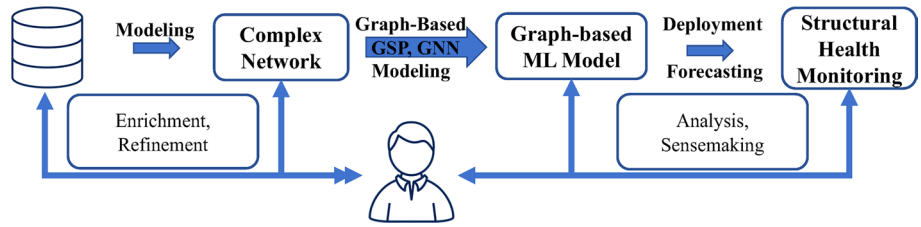


Fig. 2 A brief overview and description of the proposed Graph Signal Processing and Graph Neural Network framework for Structural Health Monitoring

where H^i is the feature representation of each node at layer $i+1$, σ the activation function (e.g., Tanh or ReLU), W^i the weights at layer i, H^i the feature representation at layer i and b^i the bias at layer i.

Kipf and Welling (2017) formalized the propagation rule in a GCN as:

$$H^{i+1} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \tag{4}$$

where W^l is the weight matrix, σ the activation function (e.g., ReLU) and $\hat{A}$ the normalized adjacency matrix with the addition of the identity matrix I and multiplying by the inverse degree matrix $\hat{D}$ of $\hat{A}$. These adaptions of the adjacency matrix and the degree matrix are necessary because of two reasons:

1. If we would multiply with the normal adjacency matrix A, then for every node, we would sum up all the neighboring nodes except the node itself. Adding the identity matrix I to A will ensure that the node features of the node in question will also be taken into account.
2. If the adjacency matrix A would not be normalized, then nodes with a high degree will change the scale of the feature vectors. Once we use the symmetrically normalized adjacency matrix $D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$, this problem is solved and the average of the neighboring nodes is used.

The resulting representation is a vector-form that can be directly used for several tasks. For example, the features can be used to predict the labels of specific nodes in a graph. Another example is applying classification of the entire graph. In this paper, we use the information to improve the results of forecasting the strain at each node in the sensor network.

Method

This section first provides an outline of our analysis framework. Then, the dataset and network modeling techniques are explained, before we describe the respective GSP and GNN methods.

Overview: GSP methodological framework

An overview on the proposed computational framework for modeling complex sensor network data using GSP/GNN in SHM is given in Fig. 2; the figure depicts the overall processing and modeling pipeline of the framework. It is easy to see, that the

presented framework provides an incremental and iterative workflow and methodology with a human-in-the-loop. For the respective steps, of the framework, we proceed as follows:

1. *Input data:* We start with the data as input for the *modeling step*.
2. *Modeling:* In the modeling step, we abstract the complex signal (i. e., the sensor network data) into a *complex network* representation.
3. Already, at this step, the complex network model can be evaluated semi-automatically in order to add refinements and adaptations of the network, for example, when faulty sensor (data) is detected.
4. *Learning/Modeling:* Next, we apply GSP and GNN learning on our network model, in order to obtain a graph-based machine learning (ML) model.
5. The resulting model can then be deployed for analysis and forecasting, in the context of SHM. Example applications include the identification of a minimal sensor subset, the detection of specific patterns, events, or mode shapes, as well as predicting specific diagnostic values—e. g., strain etc.

Below, we will exemplify the application of this framework in more detail in our case study.

Dataset

The *InfraWatch* project investigated the *Hollandse Brug* (built in 1969), a large highway bridge in the Netherlands that connects the provinces of Noord-Holland and Flevoland (Knobbe et al. 2010). After reports indicated that the bridge did not meet the quality and security requirements, sensors were placed at several locations on the bridge. This network of sensors includes 145 sensors, which contains 20 temperature, 41 vertical strain (Y-strain), 50 horizontal strain (X-strain) and 34 vibration sensors. Various data mining techniques have been applied to the dataset, including time series analysis (Vespier et al. 2012, 2013) and modal analysis (Miao et al. 2013).

The dataset that was made available to us includes 5 min of sensor data collected in high-resolution, approximately 30,000 observations in total. The original provided data was sampled at 100Hz. For smoothing the signal, we took the averaged values per 100ms. The data consists of several traffic events, where the 10 most significant are examined in this paper.

Our domain specialist suggested that the strain sensors were not measured on the same scale or at the same time. Since time synchronization is in general a challenging task when gathering simultaneous sensor data (Mechitov et al. 2004), the clock times were matched by comparing the sensor reading peaks. Afterwards, the data was normalized by rescaling them using a standard z-score standardization method.

The sensors were mounted at three different cross-sections within a single span, cf. Fig. 3 (see Miao 2014 for more visual information). As a result, in order to make the network links relevant, the 31 sensors in the middle and right cross-sections were removed. In addition, four sensors were discovered to be unreliable, reducing the total number of sensors from 145 to 110.

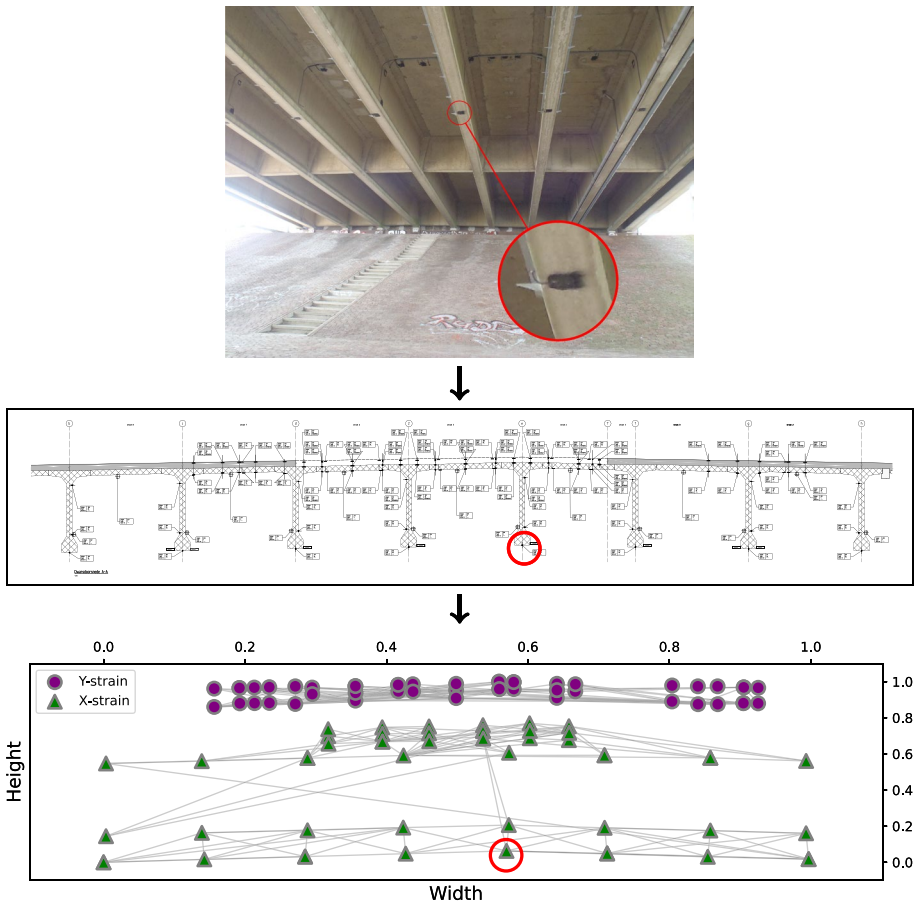
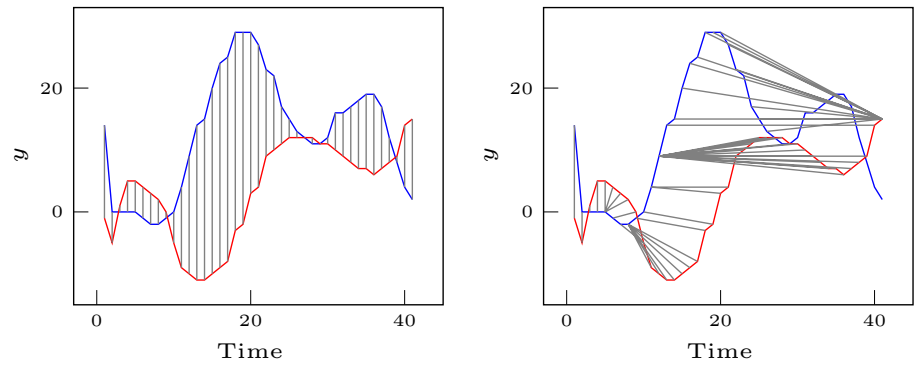


Fig. 3 Overview of how the sensor network is created. First, the photo on top shows the sensors placed on the girders underneath the bridge. This is all made visible in the blueprint figure in the middle. Lastly, the exact locations of the sensors on the blueprint were used to create the sensor network. The same sensor is highlighted in each visualized step of the procedure for clarification. The width and height of the plot at the bottom of the figure are the 0-1 rescaled locations of the sensors on the blueprint

Network creation

The blueprint of the bridge provides the geographical locations of the sensors to create the network (see Fig. 3 for the procedure). The choice for each edge (i, j) is a bit more difficult, but also a crucial step (Mateos et al. 2019). A possible direction could be using geographical distance, but that would not grasp the functional relationship between the sensors, since the girders on the bridge should catch most of the strain. As a result, although the sensors at the top of the bridge are geographically similar to the other sensors, they should behave in the exact opposite manner as the strain sensors at the girders. Therefore, the edges were determined by either (1) the correlation score or (2) the Dynamic Time Warp (DTW) distance between the sensor readings. Lastly, only the top three edges with the highest weight were added to the network (excluding the vibrations sensors, which had few edges in the first place).

The DTW distance can be calculated by first dividing time series 1 and time series 2 into equal points. Afterwards, the euclidean distance is calculated between each point in the first time series and each point in the second, where the minimum distance is stored.



(a) Euclidean Distance (b) DTW-distance

Fig. 4 Difference between the euclidean distance and the DTW-distance. The DTW method allows more flexibility in measuring time series that do not sync perfectly in time

This procedure is repeated for every point in the first time series until all data points are evaluated. The sum of all the minimum distances is then the measure of similarity between the two series (Fig. 4).

It is interesting to investigate which of the two techniques is most suited for each method used in this paper. To start, the main difference between both techniques is that DTW assumes that each time series is on the same scale, while correlation is scale-invariant. On the other hand, correlation is a more global-based measure, which means that information signalling direction (one time series causing effect in the other) is not available. Therefore, each technique has advantages in different situations over the other.

To conclude, several networks are created. First, the X-strain network with 42 sensors and 126 edges. Second, the Y-strain network with 37 sensors and 111 edges. Third, a combination of X and Y sensors with 79 nodes and 237 edges. Lastly, the vibration sensors form an especially small graph with 15 nodes and 26 edges. Each network had their own contribution to the analysis of this paper. For example, the strain sensors were used to conduct the sampling, mode shape identification and forecasting. The vibration sensors mainly assisted with identification of the mode shapes.

Node subset selection—sensor subset sampling

One of the core tasks in GSP is to "reconstruct" the signal of sensors, i. e., deducing those given a specific sample. As an example, consider a case in which cost, battery or bandwidth limits restrict the number of applied sensor nodes. In this work, we apply sampling by calculating the optimal subset of sensors that are able to reconstruct the original signal at specific time points. These specific moments in time refer to the traffic events that happen on the bridge, since calculating the error during no traffic would inflate our reconstruction error. Figure 5 motivates this decision, where the time points outside of the peaks show little deviation at all. Incorporating these time points in the evaluation will highly influence and skew the results of each algorithm.

Since brute-force searching for the most optimal solution is not feasible with a large number of N nodes, the following strategies are investigated: random search and top-down or bottom-up hill-climbing. The last two strategies are well known greedy-search

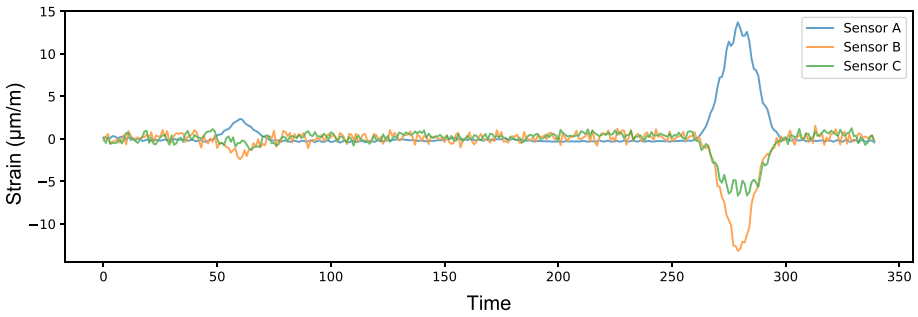


Fig. 5 Strain values (μm/m) of three sensors attached to the bridge at different locations. Sensor B and C show how the strain behaviour is influenced by their position on the bridge. Sensor A is placed at the right-most girder on the bridge; it captures most of the strain, whereas Sensor B and C are placed on the deck of the bridge where the strain is countered

strategies (Krause et al. 2008; Aggarwal et al. 2017; Anis et al. 2016; Puy et al. 2018). The random search strategy serves as a baseline and creates a random set of sensors that are sampled. On the contrary, the hill climbing algorithms either perform subset selection in a bottom-up or top-down manner. Bottom-up hill climbing (i.e., Forward Selection) starts with zero selected sensors and progressively selects the most informative sensors that decrease the error the most. Top-down hill climbing (i.e., Backward Elimination) consist of starting with all sensors and eliminating the least informative sensors one-by-one. Both hill climbing techniques incorporate randomness by choosing from the top-3 either best or worst performing sensors, which prevents the hill climbers to reach local maxima and minima. Each algorithm ran a total of 500 iterations to find the best solution and was terminated when 25% of the sensors were selected. It could then occur that the same result is repeatedly found, so only the unique solutions were stored.

To estimate the original signal from this subset of sensors, Tikhonov Minimization is applied in each iteration of the sampling procedure (Shuman et al. 2013; Defferrard et al. 2014). The function solves for the unknown vector x:

$$\arg\min_x ||Mx - y||\frac{2}{2} + \tau x^T Lx, \tag{5}$$

if $\tau > 0$ and

$$\arg\min_x x^T Lx : y = Mx, \tag{6}$$

otherwise, with the graph signal y, the masking vector M that resembles a binary vector of which nodes are sampled (1 = sampled, 0 = not sampled), the Laplacian matrix L and the regularization parameter τ. Several values for the regularization parameter in the Tikhonov Minimization were tried, of which the default value of $\tau = 0$ showed the optimum results.

Finally, each algorithm was also applied on data obtained by applying a low-pass filter $g(x)$ on the graph frequencies x of the signals:

$$g(x) = \frac{1}{1 + 0.5 \cdot x}.$$

Essentially, a graph filter performs a transformation as a function over the graph frequencies (in our example $g(x)$); it alters their contents by a point-wise multiplication in the graph Fourier domain (Isufi 2019). After the filter has been applied, the Inverse Graph Fourier Transformation of the Fourier domain signal reverts the signal back to the time domain for evaluation.

Strain forecasting method

In order to forecast the strain values in the sensor data, we applied the T-GCN utilizing the implementation contained in the Stellargraph Package (Data61 2018). The T-GCN is a Spatio-Temporal Graph Convolutional Network that combines graph layers with Long Short-Term Memory layers (Zhao et al. 2019). The spatial aspect of the data lies in the exact locations of the sensors, whereas the temporal aspect lies in the fact that different loads over time produce different stress on the bridge. For example, there could be a daily pattern in the direction of the traffic on the bridge.

The strain data of each sensor type was cut into 2914 sequences of length 10 (10 x 100ms), where the task was to predict the strain value at the 12th timestep in the future (i. e., the most difficult setting in the original T-GCN paper by Zhao et al. (2019)). In other words, we estimate the values 1.2 s later based on the preceding 1 s of strain observations. In our experimentation, we used an 80/20 split for training and testing. The model used the $N \times N$ adjacency matrix and the $N \times T$ feature matrix X, which describes the strain over T timesteps for N sensors. In this way, we regard the strain values $X_t \in R^{N \times T}$ the strain measured at each sensor on the bridge at time i. We can thus consider our problem definition as learning the function f on the network topology of the sensor graph G and the feature matrix X to calculate the strain at timestep t.

For setting up the T-GCN, two GCN layers were used with each 8 filters. These were attached to two Long Short-Term Memory (LSTM) layers with each 50 filters. LSTM layers are special recurrent layers where the top horizontal line C_t is the memory state, enabling the LSTM to remember information from the past. It also contains gates that allow or block information in the network from passing by, and these gates consist of Sigmoid functions and multiplication operations. For example, the first sigma gate in Fig. 6 functions as a *Forget Gate*, blocking or allowing information to flow through. The second sigma functions as an input gate and the third sigma functions as the output gate. Lastly, the dense output layers consist of the N sensors in the graph with Tanh as the activation function, since the strain values can be negative and fall between [-1,1]. To conclude, Table 1 shows an overview of the T-GCN model. To calculate the quality of the predictions, the Root Mean Squared Error (RMSE) of each segment is taken and compared to a benchmark taking the most recently observed value. Such a benchmark is tough to beat, since it is not expected that the strain signal will differ significantly in a short period of time.

Results and discussion

Below, we present and analyze the results of the sensor sampling from which the total signal can be reconstructed. Then, mode shape detection and forecasting applications will be discussed.

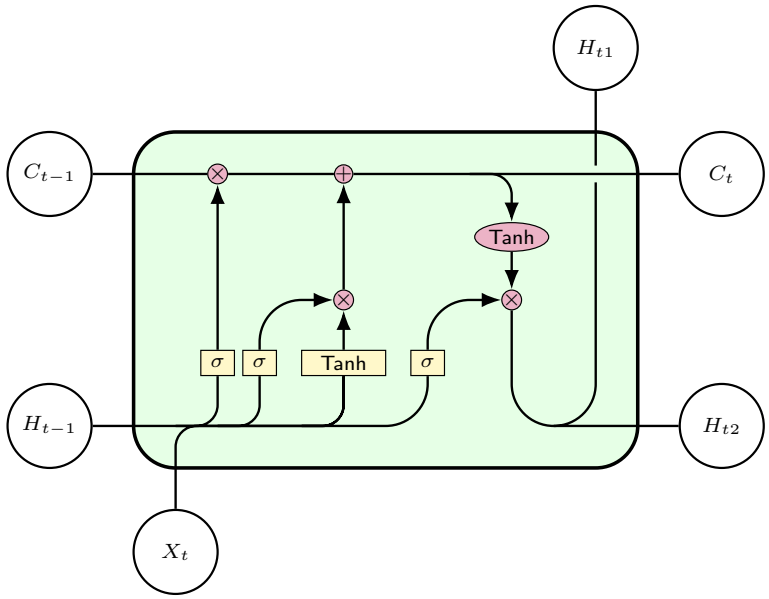


Fig. 6 Schematic overview of a single LSTM cell

Table 1 Schematic overview of the architecture of the T-GCN on the X-strain data with 42 nodes

Layer	Activation	Filters	Shape	Parameters
Input	–	–	42 × 10	0
GCN 1	ReLU	8	42 × 8	1886
GCN 2	ReLU	8	42 × 8	1870
Reshape 1	–	–	42 × 8 × 1	0
Permute	–	–	8 × 42 × 1	0
Reshape 2	–	–	8 × 42	0
LSTM 1	Tanh	50	8 × 50	18600
LSTM 2	Tanh	50	50	20200
Dropout	–	–	50	0
Dense	Tanh	42	42	2142

Sampling: selecting a minimal subset of sensors

To select the minimal subset of sensors, we used three algorithms: (1) random selection and (2) bottom-up or (3) top-down hill-climbers and assessed their performance based on the Root Mean Squared Error (RMSE) scores. We chose RMSE as a metric since it (1) penalizes errors more than other metrics and (2) measures in the same unit as the variable of interest. Each algorithm was also tested on either the correlation-based graph or the DTW-based graph during the most noticeable traffic events (see Table 2).

In terms of RMSE, the top-down algorithm continuously beats the random (+ 28.39%) and bottom-up (+ 11.41%) algorithms. In addition, the random algorithm was tested for 50.000 iterations in a separate experiment (100x the original setting). Even after so many runs, the random algorithm did not outperform both hill-climbers

Table 2 Mean and standard deviation (in italics) of RMSE scores (best performing in bold) in the DTW-graph and correlation-graph for all traffic occurrences for every algorithm

Algorithm	Sensor type					
	Non-filtered			Filtered		
	X-strain	Y-strain	Combined	X-strain	Y-strain	Combined
Correlation						
Random	0.80	1.36	1.12	0.45	0.86	0.68
	(*.32*)	(*.95*)	(*.76*)	(.29)	(*.62*)	(*.46*)
Top-down	**0.60**	**1.06**	**0.74**	**0.31**	**0.66**	**0.38**
	(*.24*)	(*.75*)	(*.52*)	(*.19*)	(*.51*)	(*.29*)
Bottom-up	0.68	1.08	0.88	0.34	0.71	0.46
	(*.30*)	(*.80*)	(*.68*)	(*.21*)	(*.53*)	(*.35*)
DTW						
Random	1.07	0.98	1.09	0.47	0.50	0.48
	(*.66*)	(*.34*)	(*.41*)	(*.31*)	(*.22*)	(*.22*)
Top-down	**0.76**	**0.76**	**0.64**	**0.31**	**0.37**	**0.29**
	(*.38*)	(*.24*)	(*.27*)	(*.20*)	(*.16*)	(*.16*)
Bottom-up	0.99	0.90	0.83	0.42	0.42	0.35
	(*.65*)	(*.29*)	(*.37*)	(*.32*)	(*.16*)	(*.18*)

The columns non-filtered and filtered show whether or not graph signal filtering was used

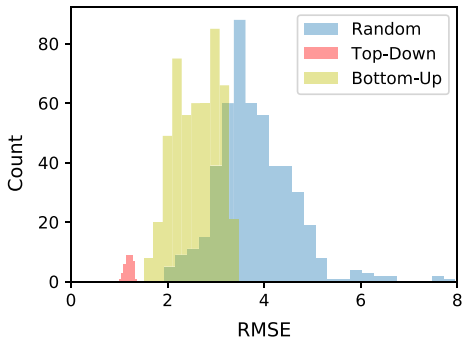


Fig. 7 RMSE scores of each algorithm on X-strain sensors during traffic event 1. in the correlation graph

when individual events were considered. In this way, a *single* top-down iteration outperforms a large number of random iterations (for any reasonable amount of iterations).

Considering the type of sensors, our domain specialist indicated that the bridge can travel more freely in the Y-direction. Therefore, modeling the Y-strain could be more difficult than the X-strain. Our results support this intuition, showing that the algorithms work well with X-strain sensors but show weaker performance with Y-strain sensors.

The DTW-based graph outperforms (+ 10.5%) the correlation-based graph in terms of overall subset recovery performance. In addition, the DTW-graph also shows a reduction in the standard deviation in the RMSE results of the Y-strain and Combined sensors. However, it is remarkable to see that while the correlation-based

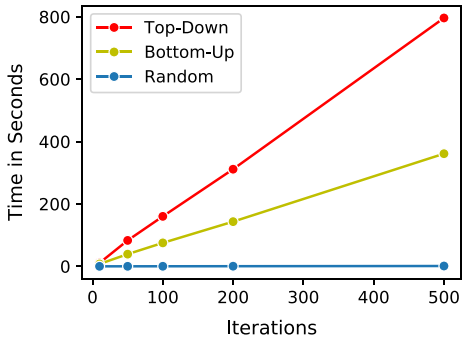


Fig. 8 Runtimes of each algorithm on X-strain sensors in the correlation graph

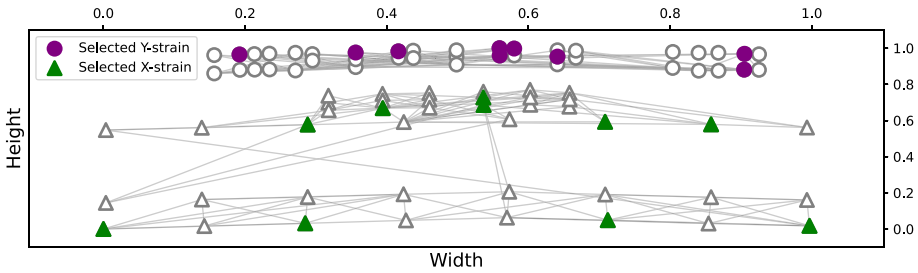


Fig. 9 For both the X-strain and Y-strain sensors, the colored nodes are the sensors selected by the top-down algorithm

graph version scores best in the X-strain condition, such a pattern is not visible in the DTW-graph. In the DTW-graph, combining the X and Y-strain sensors yields the best results.

Considering the individual performance of each subset sampling algorithm, Figs. 7 and 8 show the performance on the X-strain sensors of a specific traffic event in the correlation-based graph. Set side by side, the top-down algorithm shows the best general performance ($M = 1.20$, $SD = .06$) and shares no overlap with the iterations of the bottom-up algorithm. The bottom-up algorithm, however, already improved from the random algorithm ($M = 2.60$, $SD = .45$), which performed worst.

The top-down algorithm also has a substantially lower standard deviation, indicating that it performs more consistently when multiple iterations are carried out. The fundamental procedures of the hill-climbers will shed light on such results. The bottom-up algorithm will calculate more unique iterations since the algorithm selects sensors instead of dropping them. It calculates the 25% selected sensors, while the top-down algorithm calculates the 75% sensors *not* selected. Therefore, weak sensors will almost always be removed in the top-down algorithm whereas such sensors could potentially still remain in the bottom-up algorithm *longer*. Therefore, running a few top-down trials to determine the ideal subset seems most optimal.

When examining the selected sensors by the top-down algorithm, Fig. 9 shows a nearly symmetrical set of sensors. Such a pattern is specifically noticeable in the X-strain sensors. These results point to potential over-engineering in the number of sensors used on the bridge. Furthermore, the second-lowest row of X-strain sensors (shown in Fig. 9)

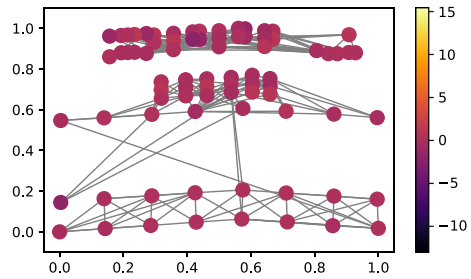


Fig. 10 The sensors network during normal conditions without traffic on the bridge

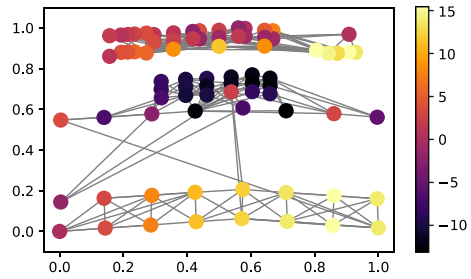


Fig. 11 The sensor network during a traffic event on the right side of the bridge

was not sampled at all. This suggests that the sensors mounted in the center of the girders are not very useful, and could potentially be left out when designing future Condition Health Monitoring projects of bridges. Perhaps engineers could use the insights from such subset sampling techniques to determine the locations of the sensors in a data-driven way in future SHM projects.

Network representation example: girders and deck

For the network representation, we depict the sensors regarding X-strain and Y-strain in Fig. 10; Fig. 11 shows the respective sensors in one visualization. When examining Fig. 10, most of the connections in the network are between the strain sensors placed at either the top or the bottom of the bridge. Such a behavior is expected, which confirms our modeling choices: The bottom part of the bridge contains girders that carry most of the weight, which should all behave very differently from the sensors placed on top of the bridge deck. Figure 11 depicts traffic event 1, in which pressure can be seen on the bottom right side of the bridge, showing that one or more vehicle(s) passed by. The figure also shows a decline in strain located at the bridge deck, indicating that the girders are doing their function properly, according to our domain specialist. Engineers could track the signals over time and determine how the pressure and vibrations are transmitted through the bridge. Figure 11 also shows a misbehaving sensor placed in the middle of the graph, of which engineers could assess whether this behavior is expected (placed on a special spot on the bridge) or not.

Figure 12 helps us investigate the behavior of the sensor network in more detail. The total variation of each node during a traffic event relative to the sensors to which it is connected is plotted as a signal on the X-strain network. The sensors placed on the

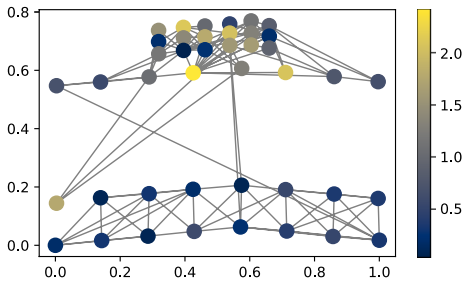


Fig. 12 Total variation during a traffic event on the X-strain sensor graph. The yellow colored nodes are the nodes whose sensor readings differ most from their neighbors

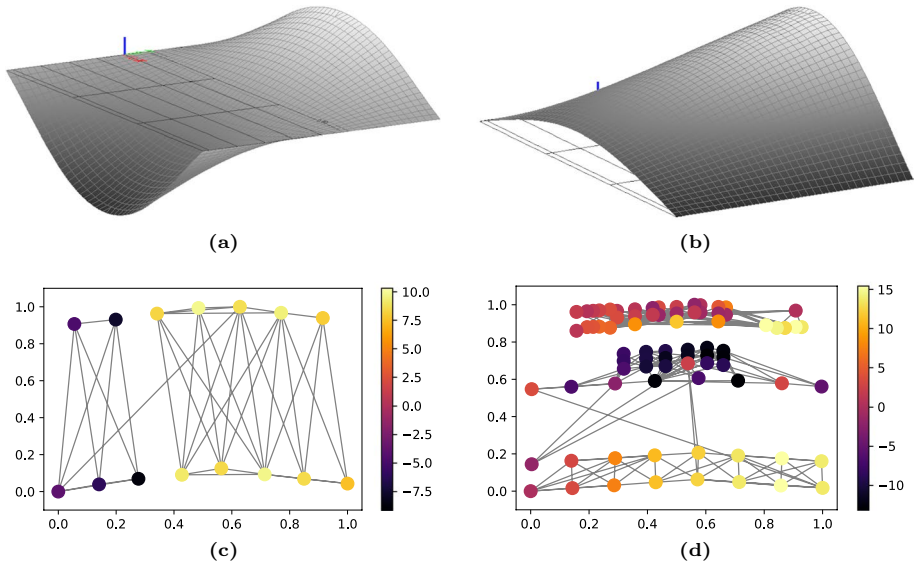


Fig. 13 **a** A FEM-based presentation of a mixture of mode shapes with the matching graph signal in **c**. **b** A FEM-based mixture of torsional mode shapes that are apparent in the girders, which are as well visible in **d**. Both **a** and **b** are from Miao et al. (2013), for which we refer to for further details on FEM

girders indicate that the strain is equally distributed across the constructed girders. However, there is a lot more variation in the strain on the deck of the bridge. The most yellow-colored sensors are the sensors that highlight this behavior. The total variation could be used in a global manner as a measure for signal smoothness, whereas the localized version could highlight inaccurate sensor readings or sensor replacement.

Identification of mode shapes

The Finite Element Method (FEM) is a computational technique for solving partial differential equations. It is commonly used to categorize the frequencies of signals into a combination of different modes in order to distinguish mode shapes. FEM can be applied for any physical phenomenon, e. g., heat flow, fluid behavior and wave propagation. FEM tries to solve a problem by partitioning a system into a set of smaller parts—the so-called "finite elements", which basically act as a representation of the entire object. Each element contains a simple equation that when combined models the global problem.

Table 3 RMSE scores (best scoring in bold) for the Graph Neural Network and Benchmark baseline on forecasting strain sensor readings in the correlation & DTW graph

Algorithm	Sensor type					
	Non-filtered			Filtered		
	X-strain	Y-strain	Combined	X-strain	Y-strain	Combined
Correlation						
Benchmark	0.49	0.79	0.58	0.36	0.59	0.45
T-GCN	**0.38**	**0.61**	**0.44**	**0.29**	**0.46**	**0.36**
DTW						
Benchmark	0.49	0.79	0.58	0.34	0.49	0.37
T-GCN	**0.38**	**0.61**	**0.44**	**0.28**	**0.39**	**0.31**

Certain mode shapes could be observed when the graph is examined for t time periods. For example, Fig. 13a shows a FEM-based mode shape, which is similar to the graph signal shown in Fig. 13c. The bridge is vibrating back and forth during this event, of which Fig. 13c highlights a left-sided decrease in vibration. Figure 13d shows a vehicle passing by on the right side of the bridge, and how the girders carry the weight and allow the other parts of the bridge to decrease in strain level, relating to the FEM-based modeling shown in Fig. 13b. A supplementary page[1] with animated GIFs is available to provide additional insights into the graph signals.

Forecasting strain and vibration

The results of the forecast with T-GCN in terms of RMSE are visible in Table 3. Overall, the T-GCN outperformed the benchmark (last observed value) in terms of RMSE by around 21%. It is interesting to see that the forecasting scores follow a similar pattern as the node subset sampling results of the correlation-based graph. In general, the X-strain sensors are easiest to forecast, followed by combining both the X and Y-strain and only Y-strain sensors.

Considering the difference between the correlation-based graph and the DTW-based graph, not a huge disparity is found. The results in the non-filtered condition are identical, only the filtered conditions show some improvements in the Y-strain and combined sensor settings. However, it is recommended to use correlation over DTW in large graph applications, since calculating the DTW-based graph takes considerably longer than the correlation graph, especially if the graph is not static and thus needs to be recalculated frequently.

The final results of the forecasting procedure are visualized in Fig. 14. Overall, the T-GCN shows very promising results in forecasting the strain. Sensor 8 & 13 show some similar behavioural patterns in the strain readings. The T-GCN is able to precisely forecast the strain on the bridge, however, it does struggle with predicting the magnitude of the events. Sensor 34 in Fig. 14 shows the T-GCN struggling a bit with the timing of the events that occur in the sensor. Such a delay can be due to the suboptimal performance of the T-GCN on this specific sensor, or the fact that the sensors

[1] https://github.com/StefanBloemheuvel/GSP_Bridge to github page.

Table 4 Parameter reduction when using node subset sampling on the T-GCN

Situation	N parameters	Ms / epoch	RMSE
All nodes	41.170	10	0.38
25% selected	33.346 (− 19%)	7 (− 30%)	0.41 (+ 7.9%)

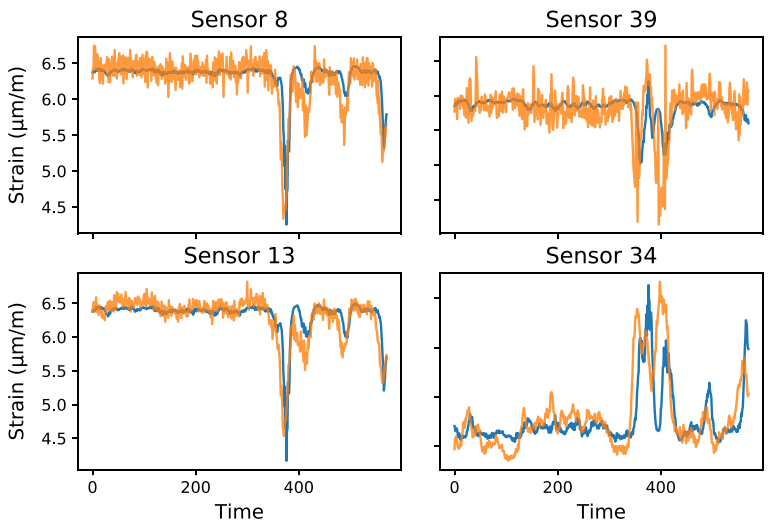


Fig. 14 Visualization of re-scaled forecasting results of a set of four strain sensors. The orange lines resemble the original signal, the blue lines the predicted values. Sensors 8, 13 and 39 show that the T-GCN is able to forecast the strain values quite well, while sometimes struggling with the magnitude of events. In addition, there seems to be some delay in the forecast results of sensor 34

had to be time-aligned (see Section 3.2). Lastly, sensor 39 shows a different behavioral pattern with fewer low-peaks in the strain of the bridge.

An interesting combination between GSP and GNNs can be conducted by applying the forecasting on the subset of nodes found by GSP. In the GSP method, the set of used nodes were reduced by 75%, leaving only 25% selected. Such a selection boils down to a parameter reduction of 19%, a Ms / epoch reduction of 30% and a (tolerable) RMSE increase of 7.9% of the selected sensors (see Table 4). Such a reduction could then also help to combat the problem that Condition Health Monitoring systems generally collect vasts amounts of data, making analysis slow (Wan and Ni 2018).

Conclusions

This work presented a computational framework using Graph Signal Processing and Graph Neural Networks for modeling complex sensor data and its respective analysis in the area of Structural Health Monitoring. That is, in this framework, we integrated Graph Signal Processing and Graph Neural Networks, covering more analytics oriented as well as more predictive/forecasting oriented techniques. In our experiments, we focused on a real-world complex sensor dataset in the context of structural health

monitoring. According to the results of our experiments with respect to the proposed frameworks and the respective approaches, both techniques revealed to be appropriate to work with the applied real-world complex sensor data.

Our results conducted on a real-world dataset indicate that GSP is capable of choosing the most essential sensors in the *Hollandse Brug*, a large bridge in the Netherlands, to derive a minimal subset of sensors from a resource-aware perspective. We also considered different strategies for network creation, investigating correlation-based and DTW-based network models. Our proposed top-down algorithm performed best of the tested alternatives in combination with the DTW-based network. With this, significant cost-reductions could be accomplished by using GSP for sensor selection in monitoring major civil infrastructures. Moreover, the sensor selection might improve the lifetime of battery-powered sensor networks, e. g., by finding the two most optimal sets of sensor to turn on interchangeably.

Furthermore, we presented a method to observe (a mixture of) mode shapes, which indicate interesting events; these results could be exploited to evaluate the condition of the bridge, since the mode shapes hint to global aspects of the bridge, such as damping and stiffness. Here, our GSP approach needs fewer modeling assumptions or background knowledge in engineering (e. g., compared to construction a FEM model).

Lastly, Graph Neural Networks were used to forecast the strain values in the bridge. The T-GCN algorithm surpassed the benchmark in each condition by around 21%. It is also interesting to note that filtering the graph signals with low-pass filters has an equal effect on reducing the forecast error as on the signal recovery using GSP. However, a possible downside of using Deep Learning is the computational complexity of such approaches. Nonetheless, the insights from the subset selection could also be used to (with a minor increase in RMSE) reduce the parameter size of the T-GCN by $\sim$20%, training speed with $\sim$30% and the amount of data with 75%.

So, in summary, in our experimentation on our real-world use case we showed that GSP enables the identification of the most important sensors, for which we investigate a set of search and optimization approaches. Furthermore, as indicated in our experiments GSP enabled the detection of specific graph signal patterns (mode shapes), capturing physical functional properties of the sensors in the applied complex network. Finally, we showed the efficacy of applying GNNs for strain prediction on this kind of data.

For future research, we intend to examine means to spot mode shapes with GSP with unsupervised techniques, e. g., by adapting methods from the field of anomaly detection (Akoglu et al. 2015; Atzmueller et al. 2017), and also to investigate Deep Learning methods in this context. Here, specifically explainable (Barredo Arrieta et al. 2020) and interpretable (Rudin 2019; Bloemheuvel et al. 2019) approaches seem interesting and relevant, e. g., building on approaches combining network-based approaches (Masiala and Atzmueller 2018) with (explainable) deep learning, in particular also for time series data (Schwenke and Atzmueller 2021a, b). This also extends to further hybrid computational approaches (e. g., Bellary et al. 2010; Barredo Arrieta et al. 2020; Dellermann et al. 2019). In addition, we intend to apply according methods using GSP and GNNs on other civil infrastructures and complex systems. For example, for potential other datasets that have already undergone sensor selection, it would be interesting to check

if our proposed techniques could reduce such datasets even further, e. g., using node subset selection capturing extended dynamics of such systems in complex contexts. This could then also help in adapting to initially hidden or emerging relationships. Moreover, alternative methods for sensor placement that do not use any graph information could be adapted to our use case and compared against our graph-based approaches. Examples are the eigenvalue vector product, the variance method and the non-optimal drive point method (Meo and Zumpano 2005) that are originally developed for optimal sensor placement instead of selection.

In addition, investigating methods of learning the network structure in a more automated way looks promising: For this, we could consider learning the Laplacian matrix, i. e., constructing the graph Laplacian from the data itself in a statistical or unsupervised manner (Egilmez et al. 2016; Dong et al. 2016). This can then aid in supporting the many modeling decisions that had to be taken in order to create the graph, which are often not easy to define (Stankovic et al. 2020). Then, it would also be interesting to compare the performance of (1) our model, (2) more semi-automatic versions, (3) and completely automatic versions for obtaining the network structure.

Abbreviations

GSP: Graph signal processing; CNN: Convolutional neural network; SHM: Structural health monitoring; GNN: Graph neural network; GCN: Graph convolutional network; T-GCN: Temporal graph convolutional network; RMSE: Root mean squared error; DTW: Dynamic time warping; FEM: Finite element method; LSTM: Long short-term memory; GFT: Graph Fourier transform.

Acknowledgements

We thank Dr. A.J. Knobbe for assisting with his domain knowledge that he gathered during managing the InfraWatch project.

Authors' contributions

SB and MA conceived of the idea and study, as well as the interpretation of the data. SB, MA and JH drafted the manuscript. SB implemented the methods and algorithms supported by JH, and ran the experiments. All authors read and approved the final manuscript.

Funding

Open Access funding enabled and organized by Projekt DEAL. This work has been funded by the Interreg North-West Europe program (Interreg NWE), project Di-Plast—Digital Circular Economy for the Plastics Industry (NWE729).

Availability of data and materials

Not applicable.

Declarations

Competing interests

None of the authors have any competing interests in the manuscript.

Author details

[1]Tilburg University, Tilburg, The Netherlands. [2]Jheronimus Academy of Data Science, 's-Hertogenbosch, The Netherlands. [3]Semantic Information Systems Group, Osnabrück University, Osnabrück, Germany.

Received: 6 May 2021 Accepted: 6 October 2021
Published online: 23 December 2021

References

Abdulkarem M, Samsudin K, Rokhani FZ, MF A Rasid (2020) Wireless sensor network for structural health monitoring: a contemporary review of technologies, challenges, and future direction. Struct Health Monit 19(3):693–735

Aggarwal CC, Bar-Noy A, Shamoun S (2017) On sensor selection in linked information networks. Comput Netw 126:100–113

Akoglu L, Tong H, Koutra D (2015) Graph based anomaly detection and description. Data Min Knowl Disc 29(3):626–688

Amaral LA, Ottino JM (2004) Complex networks. Eur Phys J B 38(2):147–162

Anis A, Gadde A, Ortega A (2016) Efficient sampling set selection for bandlimited graph signals using graph spectral proxies. IEEE Trans Signal Process 64(14):3775–3789
Atzmueller M (2014) Data mining on social interaction networks. J Data Min Digit Human 1:66
Atzmueller M, Arnu D, Schmidt A (2017) anomaly detection and structural analysis in industrial production environments. In: Proceedings of the international data science conference (IDSC 2017), Salzburg, Austria
Barredo Arrieta A, Díaz-Rodríguez N, Del Ser J, Bennetot A, Tabik S, Barbado A, Garcia S, Gil-Lopez S, Molina D, Benjamins R, Chatila R, Herrera F (2020) Explainable artificial intelligence (xai): concepts, taxonomies, opportunities and challenges toward responsible ai. Inf Fus 58:82–115
Bellary J, Peyakunta B, Konetigari S (2010) Hybrid machine learning approach in data mining. In: Proceedings of the international conference on machine learning and computing. IEEE, pp 305–308
Bloemheuvel S, Kloepper B, Atzmueller M (2019) Graph summarization for computational sensemaking on complex industrial event logs. In: Proceedings of the workshop on methods for interpretation of industrial event logs, international conference on business process management, Vienna, Austria
Bloemheuvel S, van den Hoogen J, Atzmueller M (2020) Graph signal processing on complex networks for structural health monitoring. In: Proceedings of the international conference on complex networks and their applications. Springer, pp 249–261
Boccaletti S, Latora V, Moreno Y, Chavez M, Hwang DU (2006) Complex networks: structure and dynamics. Phys Rep 424(4–5):175–308
Capellari G, Chatzi E, Mariani S (2018) Cost-benefit optimization of structural health monitoring sensor networks. Sensors 18(7):2174
Cheung M, Shi J, Wright O, Jiang LY, Liu X, Moura JM (2020) Graph signal processing and deep learning: convolution, pooling, and topology. IEEE Signal Process Mag 37(6):139–149
Clemente P (2020) Monitoring and evaluation of bridges: lessons from the Polcevera viaduct collapse in Italy. J Civ Struct Health Monit 10(2):177–182
Cornwell P, Farrar CR, Doebling SW, Sohn H (1999) Environmental variability of modal properties. Exp Techn 23(6):45–48
Data61 C (2018) Stellargraph machine learning library. https://github.com/stellargraph/stellargraph
Defferrard M, Martin L, Pena R, Perraudin N (2014) Pygsp: graph signal processing in python. https://doi.org/10.5281/zenodo.1003157, https://github.com/epfl-lts2/pygsp/
Dellermann D, Ebel P, Söllner M, Leimeister JM (2019) Hybrid intelligence. Bus Inf Syst Eng 61(5):637–643
Dong X, Thanou D, Frossard P, Vandergheynst P (2016) Learning Laplacian matrix in smooth graph signal representations. IEEE Trans Signal Process 64(23):6160–6173
Egilmez HE, Pavez E, Ortega A (2016) Graph learning from data under structural and Laplacian constraints. arXiv preprint arXiv:161105181
Gilmer J, Schoenholz SS, Riley PF, Vinyals O, Dahl GE (2017) Neural message passing for quantum chemistry. In: Proceedings of the international conference on machine learning (PMLR), pp 1263–1272
Interdonato R, Atzmueller M, Gaito S, Kanawati R, Largeron C, Sala A (2019) Feature-rich networks: going beyond complex network topologies. Appl Netw Sci 4(4):66
Isufi E (2019) Graph-time signal processing: filtering and sampling strategies. PhD thesis, Doctoral Thesis. Technische Universiteit Delft
Kipf TN, Welling M (2017) Semi-supervised classification with graph convolutional networks. In: 5th International conference on learning representations (ICLR)
Knobbe A, Blockeel H, Koopman A, Calders T, Obladen B, Bosma C, Galenkamp H, Koenders E, Kok J (2010) Infrawatch: data management of large systems for monitoring infrastructural performance. In: Proceedings of the international symposium on intelligent data analysis. Springer, pp 91–102
Krause A, Singh A, Guestrin C (2008) Near-optimal sensor placements in Gaussian processes: theory, efficient algorithms and empirical studies. JMLR 9(Feb):235–284
Lynch JP, Loh KJ (2006) A summary review of wireless sensors and sensor networks for structural health monitoring. Shock Vib Dig 38(2):91–130
Masiala S, Atzmueller M (2018) First perspectives on explanation in complex network analysis. In: Proceedings of the BNAIC. Jheronimus Academy of Data Science, Den Bosch
Mateos G, Segarra S, Marques AG, Ribeiro A (2019) Connecting the dots: identifying network structure via graph signal processing. IEEE Signal Process Mag 36(3):16–43
Mechitov K, Kim W, Agha G, Nagayama T (2004) High-frequency distributed sensing for structure monitoring. In: Proceedings of the first international workshop on networked sensing systems (INSS 04), pp 101–105
Meo M, Zumpano G (2005) On the optimal sensor placement techniques for a bridge structure. Eng Struct 27(10):1488–1497
Miao S (2014) Structural health monitoring meets data mining. PhD thesis, Doctoral Thesis. Leiden Institute of Advances Computer Science
Miao S, Veerman R, Koenders E, Knobbe A (2013) Modal analysis of a concrete highway bridge: Structural calculations and vibration-based results. In: Proceedings of the conference on structural health monitoring of intelligent infrastructure, Hongkong
Mitzlaff F, Atzmueller M, Hotho A, Stumme G (2014) The social distributional hypothesis. J Soc Netw Anal Min 4(216):1–14
Ortega A, Frossard P, Kovačević J, Moura JM, Vandergheynst P (2018) Graph signal processing: overview, challenges, and applications. Proc IEEE 106(5):808–828
Puy G, Tremblay N, Gribonval R, Vandergheynst P (2018) Random sampling of bandlimited signals on graphs. Appl Computl Harmon Anal 44(2):446–475
Rudin C (2019) Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat Mach Intell 1:206–215
Ruiz L, Gama F, Ribeiro A (2021) Graph neural networks: architectures, stability, and transferability. In: Proceedings of the IEEE

Sandryhaila A, Moura JM (2014) Discrete signal processing on graphs: frequency analysis. IEEE Trans Signal Process 62(12):3042–3054
Schwenke L, Atzmueller M (2021a) Constructing global coherence representations: identifying interpretability and coherences of transformer attention in time series data. In: Proceedings of the IEEE international conference on data science and advanced analytics. IEEE, pp 1–12
Schwenke L, Atzmueller M (2021b) Show me what you're looking for: visualizing abstracted transformer attention for enhancing their local interpretability on time series data. In: Proceedings of the 34th international florida artificial intelligence research society conference, FLAIRS/Florida Online Journals, North Miami Beach, FL, USA
Seo J, Hu JW, Lee J (2016) Summary review of structural health monitoring applications for highway bridges. J Perform Constr Facil 30(4):04015072
Shuman DI, Narang SK, Frossard P, Ortega A, Vandergheynst P (2013) 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):83–98
Sony S, Laventure S, Sadhu A (2019) A literature review of next-generation smart sensing technology in structural health monitoring. Structl Control Health Monit 26(3):e2321
Stankovic L, Mandic D, Dakovic M, Brajovic M, Scalzo B, Constantinides T (2019a) Graph signal processing—part I: graphs, graph spectra, and spectral clustering. arXiv preprint arXiv:190703467
Stankovic L, Mandic DP, Dakovic M, Kisil I, Sejdic E, Constantinides AG (2019) Understanding the basis of graph signal processing via an intuitive example-driven approach [lecture notes]. IEEE Signal Process Mag 36(6):133–145
Stankovic L, Mandic D, Dakovic M, Brajovic M, Scalzo B, Li S, Constantinides AG (2020) Graph signal processing–part III: machine learning on graphs, from graph topology to applications. arXiv preprint arXiv:200100426
Strogatz SH (2001) Exploring complex networks. Nature 410(6825):268–276
Vespier U, Knobbe A, Nijssen S, Vanschoren J (2012) Mdl-based analysis of time series at multiple time-scales. In: ECML PKDD. Springer, pp 371–386
Vespier U, Nijssen S, Knobbe A (2013) Mining characteristic multi-scale motifs in sensor-based time series. In: Proceedings of the 22nd ACM international conference on information & knowledge management, pp 2393–2398
Wan HP, Ni YQ (2018) Bayesian modeling approach for forecast of structural stress response using structural health monitoring data. J Struct Eng 144(9):04018130
Worden K (2021) Towards population-based structural health monitoring, part VI: Structures as geometry. In: Dynamics of civil structures, vol 2. Springer, pp 221–236
Wu Z, Pan S, Chen F, Long G, Zhang C, Philip SY (2020) A comprehensive survey on graph neural networks. IEEE Trans Neural Netw Learn Syst 6:66
Zhao L, Song Y, Zhang C, Liu Y, Wang P, Lin T, Deng M, Li H (2019) T-gcn: a temporal graph convolutional network for traffic prediction. IEEE Trans Intell Transp Syst 21(9):3848–3858

Publisher's Note