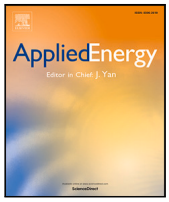
Applied Energy 353 (2024) 122151

Contents lists available at ScienceDirect

Applied Energy

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



Explainable Spatio-Temporal Graph Neural Networks for multi-site photovoltaic energy production

Alessio Verdone, Simone Scardapane, Massimo Panella *

Department of Information Engineering, Electronics and Telecommunications, University of Rome "La Sapienza", Via Eudossiana 18, Rome, 00184, Italy

ARTICLE INFO

Keywords:
Spatio-Temporal Graph Neural Network
Renewable energy sources
Time series forecasting
Smart grid
Explainability

ABSTRACT

In recent years, there has been a growing demand for renewable energy sources, which are inherently associated with a decentralized distribution and dependent on weather conditions. Their management and associated forecasting of produced energy are tasks of increasing complexity. Spatio-Temporal Graph Neural Networks have been applied in this context with excellent results, taking advantage of the correct integration of both topological data, defined by the distribution of the plants in the territory, and temporal data of the time series. A drawback of graph neural networks is the recurrent mechanism adopted to process the temporal part, which increases greatly the computational load of these models. Moreover, these models are formulated for real and sensitive contexts where, in addition to being accurate, the predictions must also be understandable by the human operator. For these reasons, in this paper we propose a novel explainable energy forecasting framework based on Spatio-Temporal Graph Neural Networks: the forecasting model generates predictions by processing temporal and spatial information using a spectral graph convolution and a 1D convolutional neural network respectively, then we apply a state-of-the-art explainer to them in order to produce explanations about the generation process. Our proposed method obtains predictions having better performance than previous approaches, both in terms of computational efficiency and prediction accuracy, with the possibility of interpreting them in order to understand the generation process. The novel approach based on fusion of forecasting and explainability in a single framework enables the creation of powerful and reliable systems suitable for real-world issues and challenges.

1. Introduction

With the advent of decarbonization processes and the implementation of energy transformation policies [1,2], shifting our energy system from one primarily reliant on economically and infrastructural efficient fossil fuels to one centered around renewable energy sources [3], the energetic sector, a crucial component in today's economy, is undergoing a profound transformation, characterized by increasing complexity and management problems [4–6]. Among the renewable energy sources, wind and photovoltaic energy emerge as two of the most prominent players, offering the promise of sustainable power generation. However, both exhibit volatility and unpredictability due to their dependence on weather conditions for electricity generation, rendering them inherently unstable and challenging to predict. The intermittent nature of wind and solar power, considered in the energy context together with the challenge of energy storage [7], requires precise supply and demand forecasting for an environmentally friendly, yet stable and resilient, energy system.

In this context, many methodologies have been developed to correctly predict the electricity profile over time produced by the plants: from purely statistical methods [8] to ensemble models [9] and systems that include meteorological modeling [10]. Artificial intelligence (AI) methodologies are having an ever-increasing success in this application field [11], starting from common statistical models (e.g. ARIMA [12]) which take into account different terms for season and trends, up to more complex deep learning systems that eliminate some of the data pre-processing steps that are typically involved with machine learning and try to automate feature extraction, removing dependencies from human experts. This is an important improvement since we can inject data from different sources directly into the system, like meteorological data for the energy production forecasting task or personal profiling data for the energy load forecasting task.

The task we consider in this paper is complicated by the need to accurately predict the flow of energy produced by all plants, over a sparse and decentralized distribution of renewable structures in the area, simultaneously and unlike previous models [13–16] that predicted the behavior of each plant singularly. In the same geographic area, energy production or weather information collected in a particular plant can be helpful to produce more precise predictions for other

* Corresponding author.
E-mail addresses: alessio.verdone@uniroma1.it (A. Verdone), simone.scardapane@uniroma1.it (S. Scardapane), massimo.panella@uniroma1.it (M. Panella).

https://doi.org/10.1016/j.apenergy.2023.122151
Received 12 March 2023; Received in revised form 1 October 2023; Accepted 21 October 2023

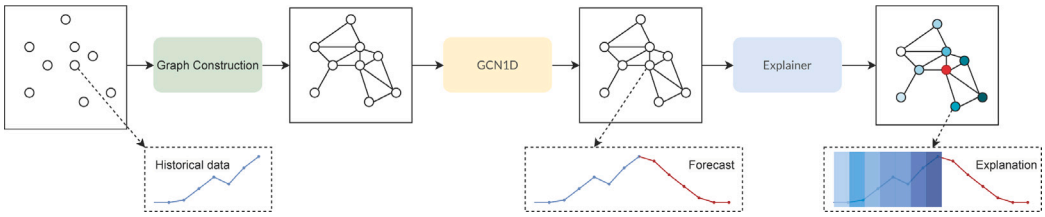


Fig. 1. This schema represents our explainable energy forecasting framework. The first step is to build the graph based on the location of the PV plants: based on the links generated, the information of the time series will be propagated among the nodes of the graph. The GCN1D forecasting model then executes a multi-step prediction for each node in the second step: in the figure, blue and red segments represent historical and predicted values respectively. Finally, the GNNExplainer generates local- and global-level explanations in order to visualize and understand the importance of historical data for the generated predictions.

plants. To assimilate as much knowledge as possible, it is necessary to correctly integrate both temporal and spatial information. Following this idea, Spatio-Temporal Graph Neural Networks (STGNNs) have been employed with success in power forecasting task [17–19], learning to predict the production output for each plant, incorporating in the process both topological and temporal information from the energy network and the time series respectively.

The ultimate goal of the development of these systems is their use in a real operational scenario, where the priorities differ slightly from the experimental environment: the complex system of electrical infrastructures in our society depends on current and even more on future decisions based on the results of such algorithms. In addition to the accuracy of the predictions, another important factor for this framework is the clarity and comprehensibility of the model's predictions to the human operator. In this context, but more broadly in deep learning frameworks, interpretability is a topic that has received increased attention in recent years: we want to try to understand black-box predictions and give human operators tools to help them do so. It is still difficult to establish in detail which attributes or features of the model's input influence a neural network's decisions. Many approaches and lines of thought have been proposed [20] to discuss the problem of what it means to explain a prediction or a model, frequently linking the concept of explainability or interpretability with the concept of the importance of some parts of the data.

The main contribution of our work is the development of a new explainable framework for the simultaneous prediction and explanation of photovoltaic (PV) energy from multiple sites by employing multivariate time series, shown in Fig. 1: the novelty of our research hinges on integrating predictive capabilities with the ability to provide explanations, both at spatial and temporal scales. In short, the primary aspects of our work can be described as follows:

- We present a novel STGNN called 'Graph Convolutional Network + 1D CNN' (GCN1D) that does not use recurrent methods to handle temporal sequences, making it more efficient and producing better results than a recurrent STGNN.
- We validate our model using multivariate synthetic time series, which have been validated in previous research [21]: we show how our model can appropriately integrate diverse time series (power, weather and timing data) in an efficient and adaptable manner.
- We describe a novel approach to incorporate explainability into traditional forecasting frameworks: we use GNNExplainer [22], a model agnostic explainability method, to explain and interpret time-varying information in an energy network.

2. Photovoltaic energy forecasting

2.1. Overview of time series forecasting

A significant amount of literature was produced for the time series forecasting problem during the years [23–25]; it is a task with a wide range of possible applications, from the financial sector [26] to the social sciences [27], from retail [28] and healthcare [29] to energy [30]. First approaches in the energy scenario relied on purely statistical methods such as ARIMA [12]. Physical models and equations able to simulate environmental conditions [31] were also used. PV power output was forecasted using numerical weather prediction (NWP) models [32]. In studies on power generation forecasting, ensemble approaches incorporating previous methodologies are used: hybrid models as in [33,34] or multistage systems as in [35] are proposed in order to achieve the defined goal, while minimizing some model's drawbacks possibly also subject to some physical or operation constraints. Recent approaches to the problem involve the use of complex frameworks such as fuzzy logic [36], genetic algorithms [37] and deep neural networks [38].

For many years, Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have defined the state of the art. Mono-dimensional CNNs (or 1D CNNs) operate on one-dimensional data sequences: they can be applied directly to raw data without any pre- or post-processing. Furthermore, simple 1D convolutions, which consist of scalar multiplications of the signal with a kernel and additions, are very computationally efficient. Many signal processing applications made extensive use of 1D CNNs [39]. RNNs are neural networks that allow previous outputs to be used as inputs while having hidden states: they excel at processing sequential data and detecting temporal trends, but they may suffer from exploding or vanishing gradients and they are also slow to train, having to backpropagate through the temporal sequence. Meteorological data are used by Liang et al. in [40] to train a model based on stacked Long Short-Term Memory (LSTM) units to forecast wind power generation, yielding more accurate predictions than a single LSTM. Hossain et al. in [41] and Succetti et al. in [42] propose both stacked LSTM network to forecast PV energy generation. Oreshkin et al. present N-BEATS networks in [43], an interpretable architecture for univariate time series forecasting, based on backward and forward residual links and fully-connected layers.

More recent approaches to the problem involve the use of recently developed techniques that have achieved great results in deep learning, such as attention mechanisms or, more specifically, transformers: in [44], an attention mechanism is used in multi-step prediction tasks while Grigsby et al. developed in [45] a Transformer-based model, called 'Spacetimeformer', applied to the multivariate time series forecasting task. FEDFormer was proposed in [46], a model which combines a seasonal-trend decomposition method with a frequency-enhanced Transformer to improve its performance for long-term prediction. Transformer-based models for long-term time series forecasting tasks are studied in [47], which proposes 'Informer', a computationally efficient model that implements a ProbSparse self-attention mechanism, and in [48] where a novel decomposition architecture, named 'Autoformer', with an Auto-Correlation mechanism was presented. Although the aforementioned works make use of temporal correlations, they do not completely utilize the spatial information that derives from the topological distribution of the plants.

Many actual PV forecasting models have a significant flaw in that they only use data from one generation system at a time, ignoring any information or data collected by neighboring power plants. Renewable

energy plants are distributed across a specific territory, and sharing information among them can improve forecasting task performance. GNNs are a promising approach for this task: they comprise a set of layers and techniques to define deep neural networks that can process data in the form of a graph structure. We want to construct node representations that are based also on the structure of the graph. GNNs have sparked a lot of interest due to their expressive capacity and ability to infer information from complex data, such as brain signals, social network connections, and traffic congestion patterns, for example in [49–51]. The task of forecasting power time series can be thought of as a node regression task, with each plant acting as a node and each temporal sequence of power values acting as a sequence of node features corresponding to the respective station.

STGNNs are the most suitable GNN category for multi-site time series forecasting tasks. This GNN family excels in handling dynamic processes with graph-like data. They have been used in the latest research in a variety of scenarios, ranging from traffic forecasting [52,53] to evaluate COVID-19 epidemiological observations [54]. Fang et al. develop in [55] a novel continuous representation of GNNs in tensor form employing ordinary differential equations able to learn long-range spatial–temporal correlations in the traffic forecasting scenario. Attention mechanism was used also with GNNs for traffic forecasting tasks: Zheng et al. propose in [56] a graph multi-attention network based on the encoder–decoder structure while Qin et al. develop in [57] a model by combining memory attention, graph convolutional network and LSTM.

Several STGNNs, which are based on the recurrent mechanism, were developed for scanning temporal patterns in graph-like data. Bai et al. propose a GCN that combines modules to capture node-specific patterns and to infer inter-dependencies among time series [58], with recurrent networks. Other STGNN models based on recurrent mechanisms, like GConvLSTM or GConvGRU [59], GC-LSTM [60], or DCRNN [61], are very effective in predicting temporal data, but, as with RNNs, the recurrent component incurs a significant computational cost, limiting their use in a real-world operational context.

2.2. Explainability approaches

Despite their success and progress in many fields of application, deep learning models are still treated as black box models: the underlying mechanism that steers the learning process and generates predictions is not fully understood. Excellent performance and accuracy are insufficient to ensure model deployment in production, particularly in critical operational contexts characterized by additional measurement performances such as robustness or safety; in these environments, the human operator must understand and trust the model's predictions, as in the healthcare, judicial systems or energy management. Explainability techniques have been used in many domains of deep learning literature, from Natural Language Processing [62] to Computer Vision [63], from multimodal models [64] to real scenarios such as medicine [65] or finance [66].

Several approaches were also developed for GNNs: Yuan et al. unifies in [67] all of the previous approaches by defining a taxonomy to classify explainability methods for the GNN framework, based on what we want to explain and which high-level tools or techniques we want to use. It divides graph neural network explanation methods into instance-level and model-level explanations: the former provides input-dependent explanations for each graph instance in input, while the latter generates GNN explanations without regard to any specific input example. A large number of developed techniques that belong to the first family of methods can be further classified into four subgroups based on what they explain and, more essential, how they explain it.

The *gradient or features-based methods* use the gradients or hidden feature map values as the measure of the degree of importance of the input: generally, higher gradients or feature values are related to the most important features. Sensitivity Analysis in [68] directly employs the squared values of gradients as the importance scores of different input features while CAM and GradCAM [69] map the node features in the final layer to the input space to identify important nodes.

Perturbations methods rely on the motivation that predictions should be similar when perturbing or modifying input information that is not relevant to the task, finding out in this way important features. Several models based on the perturbation technique have been developed lately: GNNExplainer [22] generates instance-level explanations by maximizing the mutual information between a GNN's prediction and the distribution of possible subgraph structures, identifying a subgraph and a subset of node features important for the prediction. Other important methods are PGExplainer [70], ZORRO [71] and GraphMask [72].

In *surrogate methods* a simpler and interpretable model is employed to explain the predictions generated by the main one for the neighboring areas of the input example: naturally, it is assumed that the relationships that occur can be captured by a simpler model. Surrogate models that have been developed recently are GraphLime [73], a model able to generate local interpretable model explanations using the Hilbert–Schmidt Independence Criterion, PGM-Explainer [74], and RelEx [75].

Decomposition methods split the original predictions into several terms, each of them is then considered as a score of the importance of the corresponding input features. These techniques directly examine the model parameters to identify the connections between the input space features and the prediction outputs. LRP [68], Excitation BP [69] and GNN-LRP [76] are some examples of decomposition methods. In particular, GNN-LRP explains GNNs by identifying groups of edges that jointly contribute to the prediction.

Finally, due to the complexity and indeterminacy of the goal, very few approaches for the model-level explanation family have been developed: the only existing approach of this type, which can be further classified in the subfamily of generation methods, is the XGNN [77]: it interprets GNNs at the model-level by training a graph generator so that the generated graph patterns maximize a certain prediction of the model.

3. Proposed graph-based energy forecasting framework

Classical photovoltaic energy forecasting systems have typically been applied to single energy stations: the problem has been formalized by taking into consideration only the temporal character, for this reason we refer to it as energy time series forecasting. Given the scalar time series of a photovoltaic plant $s(m)$, $m > 0$, assuming that the current sample at time index m and all of the previous ones are known, the task to be completed concerns the prediction of future values of s, i.e. $s(m + w)$, $w > 0$. To better understand the future dynamics, instead of using only the information concerning the energy production of the single plant (univariate approach), other time series s_i, typically correlated to meteorological conditions, are incorporated in the forecasting method (multivariate approach). Finally, the problem is tackled as a regression problem, employing the Mean Squared Error (MSE) as a loss function, and assessing performance with the Mean Absolute Error (MAE) and the R-squared (R^2) metric, also called coefficient of determination, by comparing predicted and real future values. To predict multiple implants simultaneously, the model would need to be adapted and rerun for each plant, which is impractical and ineffective. The proposed solution involves the use of STGNNs, which thanks to the use of spatial and relational information between PV plants allows to generate explainable and more accurate predictions for each site. We showcase our framework in Fig. 1, and we describe each component in depth in the following subsections: building the latent graph (Section 3.1), forecasting (Section 3.2), and explaining the predictions (Section 3.3).

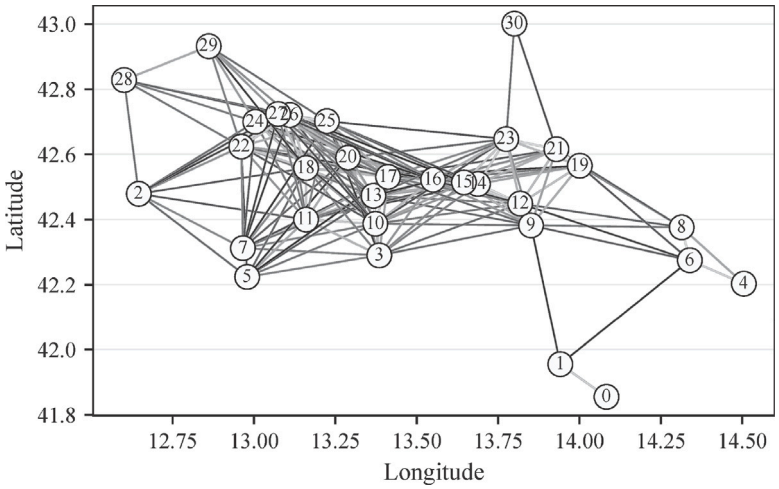


Fig. 2. Final node network after data preparation: the darker the link, the greater the distance from the two nodes it connects. Such distances are compared with a threshold during the cutting edge operation.

3.1. Graph construction

The first step concerns the construction of the graph to be used in our model. In the graph, each node corresponds to an energy plant, but the creation method of the connections between the various plants, important since it will define the flow of information during the message passing phases, is not unique. Unlike other scenarios, such as the chemical one where the relationships in a graph represent the chemical bonds, in our context there are no explicit links between the various plants. One solution is to assume that each node in the dataset is connected to every other node, thus considering a complete graph; this is obviously not a computationally efficient solution because the number of edges grows very quickly with the number of nodes as $n(n-1)/2$. In this paper we consider a simplified threshold-based algorithm inspired to [78], but we note that more sophisticated and data-driven methods such as [79], which learns a trainable function that predicts edge probabilities of the graph depending on the downstream task, exist, and we leave their analysis for future works.

From an application point of view, we define the edge creation process by setting an user-defined threshold γ, and then cutting off any edge greater than that distance, more formally: $e(x_i, x_j) = \mathbb{I}_{d(x_i,x_j)} \leq \gamma$. The value of the threshold is set as the minimum allowable distance d resulting in a connected graph. Given a complete graph, if for each node we collect its shortest link, the threshold value of the cutting operation will correspond to the maximum length present in this set. The graph obtained at the end of this procedure, as the one shown in Fig. 2, will be used for the subsequent tasks.

3.2. Forecasting

Given a set of $N > 1$ power plants distributed in a heterogeneous way in a territory, let $s_n(m)$, $m > 0$, $n \in \{1, N\}$, be the scalar power time series related to the n-th power plant. Suppose that the current sample at time index m and all of the previous ones are known, and that $s_n(m + w)$, $w > 0$, represents the samples in the future to be predicted up to distance w. In a univariate and single-site approach, this problem can be expressed as a regression problem by using past samples of the sole time series of the single station under analysis.

In our approach instead, we consider multiple signals from all the PV plants: so the information in entrance to the proposed GCN1D model can be depicted as the matrix $\mathbf{X}_\tau \in \mathbb{R}^{N\times C\times D}$ representing the previous D values for each channel C at the time step τ for N nodes. Given the future time series prediction task defined as $f_{\boldsymbol{\alpha}}(\mathbf{X}_\tau) \cong \mathbf{H}_\tau$, where $f_{\boldsymbol{\alpha}}$ is our chosen parametric estimator and $\mathbf{H}_\tau \in \mathbb{R}^{N\times W}$ the matrix of the future W values at time step τ to predict for N nodes, the learning problem consists in finding a set of parameters $\boldsymbol{\alpha}$ that minimizes the prediction error E, which is defined for each considered time step τ as:

$$E = \arg\min_{\boldsymbol{\alpha}} \sum_{\tau} \left\| f_{\boldsymbol{\alpha}}(\mathbf{X}_\tau) - \mathbf{H}_\tau \right\|_2^2 . \tag{1}$$

After the graph construction phase, a set of PV plants is represented as an undirected weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$, with a set of nodes $\mathcal{V}$, a set of edges $\mathcal{E}$, and finally a symmetric adjacency matrix $\mathbf{A}$ of size $N \times N$ which encodes the topology of the network. Indeed, if nodes v_i and v_j are connected, there will be an edge, which we define as e_{ij} and a corresponding element a_{ij} in the adjacency matrix; if an edge does not exist a_{ij} will be zero. The matrix $\mathbf{D}$ is the node's degree matrix of the graph: it is a diagonal matrix and it is constructed by definition counting for each node the number of connected edges, so $D_{ii} = \sum_j A_{ij}$. The Laplacian matrix of the graph is defined as $\mathbf{L}$: it is a positive semidefinite matrix and it can be decomposed as $\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$, where $\mathbf{U}$ is a unitary matrix of eigenvectors and $\boldsymbol{\Lambda} \in \mathbb{R}^{N\times N}$ is the diagonal matrix of associated eigenvalues λ_i, $i = 1 \dots N$.

Convolutions on graphs are historically classified as using either spatial or spectral convolutions. Spatial graph convolutions, like GraphSAGE in [80] for example, exploit the local neighborhood of the nodes up to k-hop of distance from the central node: they are simpler and more intuitive to understand. Spectral graph convolutions were introduced by Bruna et al. in [81]: they employ graph signal processing techniques to exploit the spectral components of the graph to better understand complex patterns of its structure. To do this, the eigendecomposition of the Laplacian matrix of the graph is performed, but this can be a slow process since its complexity $O\left(n^2\right)$ increases quadratically with the number of nodes. Defferrard et al. solved this problem in [82], by introducing an approximate version of the spectral graph convolution using Chebyshev polynomials: the method provides strictly localized filters and has a complexity of $O(|E|)$. The graph convolution employed in our model is based on a modified version of the Chebyshev Graph convolution.

In the following, we will describe our GCN1D model, which is used in multi-site and multi-step time series forecasting. The method presented in [82] is employed to parametrize the spectral parametric filter using the recursive formulation of Chebyshev polynomials to overcome the expensive computational cost of classical spectral convolutions. We modify the classical Chebyshev convolution operator to make it more suitable for time series data: instead of computing a linear transformation between input $\mathbf{X}$ and the weights Θ, we apply a one-dimensional convolution to the input. The graph convolution operation ($GraphConv$) can be written as:

$$y = g_\Theta *_\mathcal{G} \mathbf{X} = \sum_{k=0}^{K} \mathbf{P}_k(\tilde{\mathbf{L}})\mathrm{CNN}_{1D}(\mathbf{X}), \tag{2}$$

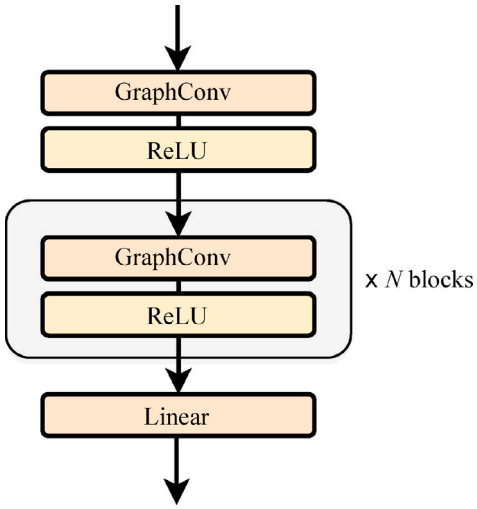


Fig. 3. Structure of the proposed GCN1D model.

where the parameters $\Theta_k \in \mathbb{R}^{C\times O}$, with C input channels and O output channels, are the trainable filters of the 1D CNN and $\mathbf{P}_k(\tilde{\mathbf{L}})$ are Chebyshev polynomials defined as:

$$\mathbf{P}_0 = 1, \quad \mathbf{P}_1 = \tilde{\mathbf{L}}, \quad \mathbf{P}_k = 2\tilde{\mathbf{L}}\mathbf{P}_{k-1} - \mathbf{P}_{k-2}\,. \tag{3}$$

The weight matrices Θ_k encode the Chebyshev coefficients of the filtering operation. So the graph convolution is applied to $\mathbf{X} \in \mathbb{R}^{N\times C\times D}$ and it returns $y \in \mathbb{R}^{N\times O\times W}$, with D and W past and future time steps. $\mathbf{P}_k(\tilde{\mathbf{L}}) \in \mathbb{R}^{N\times N}$ is the Chebyshev polynomial of order k evaluated at the scaled Laplacian $\tilde{\mathbf{L}} = 2\mathbf{L}/\Lambda_{\max} - \mathbf{I}_n$. Moreover, the graph convolutional filter represented with polynomials until the order $K-1$ of the scaled Laplacian is spatially localized and only depends on nodes that are at a maximum K-step of distance from the central node.

The 1D CNN model has been employed extensively for 1D signal processing tasks, as shown in [83]: it learns filters that correlate input and output channels of the signal and it is faster and lighter compared to other neural approaches. In order not to violate the condition that future time steps influence previous time steps, all convolutions are causal. Causal convolution is a type of convolution used for temporal data, which ensures the model cannot violate the ordering in which data are processed; i.e., the prediction $p(x_{t+1}|x_1,\dots,x_t)$ emitted by the model at time step τ cannot depend on any of the future time steps $x_{t+1},\dots,x_T$. For 1D data, causal convolution can be implemented by shifting input data: we pad only the left side of the input sequence and this ensures that the convolutional layer does not peek future time steps when making predictions.

The proposed GCN1D network can be represented as a stack of graph convolutional layers, each of them implementing the previous *GraphConv* operator. The parameters of the 1D CNN vary from one layer to another following a specific pattern: we start with a larger kernel size of dimension 5 in the first 3 layers and then we decrease it to 3, while the hidden feature dimension is decreased layer by layer, from 32 to 12, and the number of channels follows an inverse behavior, from the 5 initial multivariate channels to 256 in the final layer. The structure of the model can be seen in Fig. 3.

3.3. Explainability

Despite excellent results and a broad range of applications, STGNN models lack of clarity and transparency in their predictions from a human point of view, like most of deep learning models. So, the second main objective of this work relies on the comprehension and understanding of the predictions generated by the model. As discussed in Section 2.2, instance-level explanations are the most used techniques to be employed in explainability analysis: they give a complete description for each sample of what is important for that prediction, and we think that for a model that has as an ultimate goal the effective employment in a real scenario, this is an important fact. Providing input-dependent explanations for each graph, these methods explain GNNs by identifying input features that play a key role in the prediction.

In the multivariate setup, i.e., when using more time series information than the single time series that serves as the target (the power production), the input has a dimension $\mathbf{X} \in \mathbb{R}^{N\times C\times D}$ where N is the number of nodes, C is the number of channels in the time series, each of them representing a different signal (e.g. power, temperature, etc.) and D is the number of previous time steps that we give in input to the predictor. So this method will allow us to interpret the importance of each single time step; subsequently, this knowledge can be aggregated to draw up considerations about nodes or features concerning the task.

The GNNExplainer is a post-hoc instance-based explanation model that belongs to the perturbations-based group [22]; it is a model agnostic approach exploitable on GNN models for several graph-based tasks. Given the whole graph G and the set of node features $\mathbf{X}$, it generates explanations by finding a sub-graph G_s and a subset of node features $\mathbf{X}_s$ which are important for the problem. These elements are obtained by formulating a mean field approximation and learning trainable masks to apply via Hadamard products to both node features and edges. The optimization process is then driven by maximizing the mutual information (MI) between the real prediction Y of the GNN model and the distribution of possible subgraphs structures and node features $(G_s, \mathbf{X}_s)$ discovered by the trainable masks, by introducing the basic loss L_m:

$$L_m(h,\hat{h}) = \mathbb{H}(Y) - \mathbb{H}(Y|G = G_s, \mathbf{X} = \mathbf{X}_s)\,. \tag{4}$$

This formula captures the difference between the value of the entropy of Y with respect to the conditional entropy of the prediction generated by the minimal graph. The loss function is therefore based on the main terms $L_m(h,\hat{h})$, which represent the mutual information between the real prediction of the model and the one masked.

Other regularization terms are added in the loss function formulation in order to generate appropriate subgraphs. An element-wise binary entropy function encourages the structural and node feature masks to be as discrete as possible:

$$\mathbb{H}(m) = -m\log(m) - (1-m)\log(1-m)\,, \tag{5}$$

where m is the binary mask we want to obtain; i.e., values related to important features are closer to 1 while irrelevant ones are closer to 0.

An ℓ_1-norm term, summing all the elements of the mask parameters in order to penalize larger explanations mask size, can increase sparsity and lead to a minimal representation. Both ℓ_1-norm and entropy components are computed for $\mathbf{M_n}$ and $\mathbf{M_e}$, which are node features and edges masks, respectively. Additional parameters α_1, α_2, β_1 and β_2 are multiplied with regularization terms for both node features and edges formulation: they modulate the sparsity of the masks measuring the influences of ℓ_1-norm or entropy in the loss formulation. Node features and edges loss components are represented by the following equations:

$$L_n(\mathbf{M_n}) = \alpha_1 \left\|\mathbf{M_n}\right\| + \alpha_2 H(\mathbf{M_n})\,, \tag{6}$$

$$L_e(\mathbf{M_e}) = \beta_1 \left\|\mathbf{M_e}\right\| + \beta_2 H(\mathbf{M_e})\,. \tag{7}$$

Overall, the final loss can be formulated as:

$$L = L_m(h,\hat{h}) + L_n(\mathbf{M_n}) + L_e(\mathbf{M_e})\,. \tag{8}$$

Unfortunately, the resulting masks at the end of the training process are soft masks having continuous values and not discrete ones. They suffer from the introduced evidence problem, discussed in [84] and the potential side effects associated with the related mask operations, as

stated in [67]. These masks can be transformed into discrete masks by performing a threshold operation, then setting a value and assigning 0 or 1 to the mask values below or above the preset value, respectively. In our case, for the node features mask we generate a trainable mask with the same dimension as the input of our model; the edge mask instead will have the same dimension as the number of edges.

4. Experimental setup

4.1. Dataset and evaluation metrics

We need multivariate time series data to run our experiments and validate our forecasting and explanation models. We use synthetic time series generated by the PVGIS (Photovoltaic Geographical Information System) web interface [85], a project developed by the European Commission Joint Research Centre that provided simulated PV power output data. Solar irradiance is calculated using satellite detection, and PV power output is calculated for each location based on PV plant mounting type (fixed, moving structure with a vertical or inclined rotating axis, two axes), slope and azimuth angle (depending on mounting type), PV technology (Crystalline silicon, CIS, CdTe), installed peak PV power, and system performance loss.

In addition, meteorological time series are reconstructed in each location, with 2-meter air temperature and 10-meter total wind speed recorded. Their effectiveness in multivariate time series forecasting in comparison to real-time series has been validated in previous works: they are a viable alternative to the lack of publicly available datasets of real plants collected at the same time period that include meteorological information like wind or temperature.

Using this tool, we generated a dataset of 31 PV plants distributed irregularly across an area of 25.500 km^2; PV plants were simulated by sweeping through a large number of technical plant parameters to recreate time series that are as different as possible in order to reproduce a real scenario. Each sample is collected at a sampling rate of one hour and the total sampling period is one year; experiments were conducted using only the three-month winter time series. However, the different results pertaining to tests for each season and for the whole year will be shown in Section 5, in order to understand how the models react to different seasons.

4.2. Data preparation

The time series are preprocessed after they have been collected: each time was normalized to stay in the interval [0,1], given the minimum and maximum value of each time series. For our experiments, we use 5 different time series typologies that help the model obtain even more precise predictions. In addition to the three mentioned ones (i.e., output power, wind speed and temperature), the model also includes temporal values for the month and the hours: the information of the month allows to take into account long-term variations in the weather while the hourly information is strictly correlated to the solar production during the 24 h of the day. Furthermore, unlike temperature and wind, they do not involve any additional cost.

4.3. Experimental settings

PyTorch has been used to create the experimental setup. Graph models were created in particular using the graph-specific frameworks PyTorch Geometric [86] and PyTorch Geometric Temporal [87]. The dataset, both for real and synthetic time series, was generated by dividing the series into contiguous time samples in the range $[s(t-D:t), s(t+1:t+W)]$, which represents the D input values and the W values to predict for all nodes; training, validation and test datasets are constructed by assigning them 70%, 15% and 15% respectively of the total samples randomly.

For the forecasting task, we have compared our problem with 1-step and 24-step ahead tests, considering statistical approaches like ARIMA, non-graph neural models, LSTM [88], and other STGNN models very effective in the task of processing time-varying nodes features. These models are the GConvLSTM and GConnvGRU [59], GC-LSTM [60] and DCRNN [61]. All of these GNN models were trained with ADAM [89] optimizer using early stopping for a maximum number of 500 epochs. For the LSTM model only, the maximum number of training epochs was set to 1000. The learning rate was initially set to 0.001 and it is dropped by a factor of 0.9 when the MSE validation metric does not improve after 10 training epochs. Minibatch size was settled to 16. All the experiments have been carried out on a machine equipped with a 6 cores Intel® Core™ i5-9400F CPU @ 2.90 GHz, 16 GB of RAM and a Nvidia® RTX 2060 GPU with 6144 MB of RAM @ 1.830 GHz.

4.4. Model settings

For the LSTM-based model we employ a 6-layer LSTM model with a final fully connected layer to adapt the output dimension to the predicted window dimension; hidden feature dimension was set to 50. For the GConvLSTM, GConvGRU, GC-LSTM and DCRNN we adopt the two layer architecture as in [21], with a final fully connected layer. For all of them the Chebyshev filter size is set to 2 and the hidden feature dimension is 128. Instead, the proposed GCN1D model consists of a stack of 9 convolutional layers. Despite the bigger number of layers compared to other graph neural models, it shares a similar number of parameters. No dropout is applied to any model as it has been seen that its use worsens the predictions of the model. GNNExplainer model was also trained with ADAM optimizer for 500 epochs with a learning rate of 0.01. The weights for the node features and edge masks in the loss function, α_1, α_2, β_1 and β_2, have been respectively set to 1.0, 0.1, 0.005 and 0.1, as suggested by Ying et al. in [22].

5. Numerical results

In this section, we present the results obtained for both forecasting and explainability tests. First, we introduce results and analysis obtained in the forecasting task: our method outperforms previous methods both in terms of accuracy and computational efficiency. Then, we make several tests to show the robustness of the model to different operational scenarios, varying time series information or input and prediction windows. Finally, once demonstrated the quality of our predictions, we use GNNExplainer to generate post-hoc explanations, showing how this type of method is an important added value for an energy network. For visual clarity, MSE and MAE results in the following tables are expressed as numbers in the order of 10^{-5}, while elapsed time is measured in minutes.

5.1. Forecasting results

In the first experiments, we compare the GCN1D network with respect to other statistical and neural network approaches previously introduced in Section 4.3. The task we have conducted in the synthetic dataset is the prediction of the solar energy production in the next 24 h for each plant, receiving in input to the system the past 24 h. Input and prediction lengths will be the same for all next experiments unless otherwise specified. The numerical results are summarized in Table 1, where the number of model parameters is also reported; a possible forecasting output of our model is illustrated in Fig. 4.

We can observe how the GCN1D model overcomes all previous methods in terms of MSE, R^2 and MAE. Statistical methods produce higher errors with respect to neural approaches; LSTM achieves good error results in less time than any other approach, but it fails to reach graph-based performances. STGNNs produce excellent predictions, but the training times are very high: although the recurrent mechanism is an efficient tool to learn temporal patterns, it is affected by a

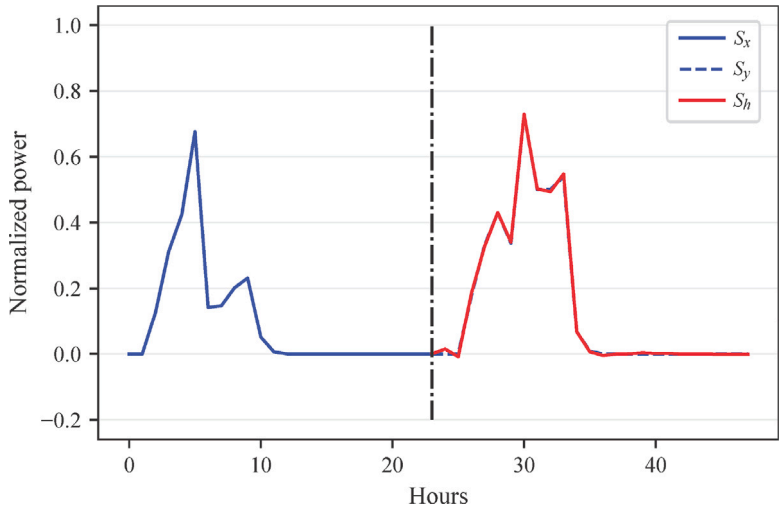


Fig. 4. 1-Day ahead forecasting result for a single node. The model is fed by the past 24 h (S_x) and it generates the prediction for the next 24 h (S_h), which is compared to the ground truth (S_y).

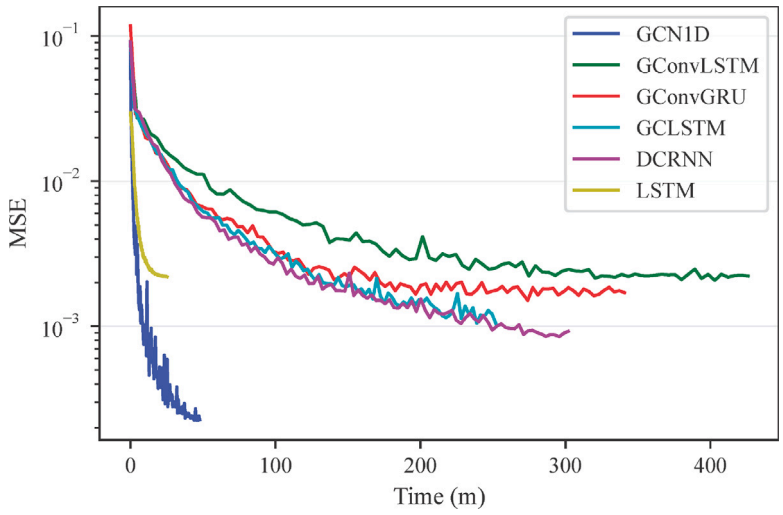


Fig. 5. Validation loss during the training for each model in log-scale; graph recurrent models are very slow with respect to GCN1D network.

Table 1
1-Day Forecasting Results for Considered Models.

Model	MSE	MAE	R^2	Parameters	Time
1-step ahead	2550	7474	0.608	–	–
24-step ahead	3947	8862	0.394	–	–
ARIMA	1645 ± 467	7241 ± 1152	0.747 ± 0.077	–	396.4
LSTM	311 ± 44	2430 ± 176	0.952 ± 0.007	142 k	22.4
GConvLSTM	162 ± 10	1965 ± 68	0.975 ± 0.002	405 k	495.89
GConvGRU	139 ± 8	1991 ± 64	0.979 ± 0.001	303 k	357.70
GC-LSTM	199 ± 44	2219 ± 194	0.969 ± 0.006	335 k	284.35
DCRNN	176 ± 32	2180 ± 301	0.973 ± 0.004	601 k	315.20
GCN1D	**105 ± 40**	**1245 ± 240**	**0.984 ± 0.006**	730 k	**45.56**

Table 2
Forecasting Results with Different Multivariate Information.

Time series	MSE	MAE	R^2
Power	323 ± 10	2520 ± 149	0.950 ± 0.002
+ temp	299 ± 27	2456 ± 166	0.954 ± 0.005
+ wind	250 ± 73	2268 ± 306	0.962 ± 0.012
+ hours	212 ± 91	2076 ± 428	0.967 ± 0.015
+ month	**105 ± 40**	**1245 ± 240**	**0.984 ± 0.015**

high computational load that disincentives an effective usage in real scenarios. Our model instead reaches the best results with a small training time compared to recurrent STGNNs.

In order to better understand the difference in terms of computational efficiency, the validation loss in log-scale of the previous models in function of the training time is shown in Fig. 5, while in Fig. 6 we show an histogram plot representing the time needed for each model to reach a fixed level of loss during the training. We can see how the GCN1D network not only achieves error levels better than the other models, it also does so in a much shorter period of training time.

Subsequently, we wanted to analyze how much the introduction of different types of information can increase the predictions. Table 2 shows the results of this experiment: progressively, starting from the univariate case, in which we use only the data relating to the electrical power of the system as input, we add temperature, wind speed, hour and monthly time series one at a time. It can be seen that the more the number of time series introduced, the lower the error, where the best result is achieved when all different time series are employed. Therefore, this experiment demonstrates how the system can correctly incorporate and benefit from different information channels.

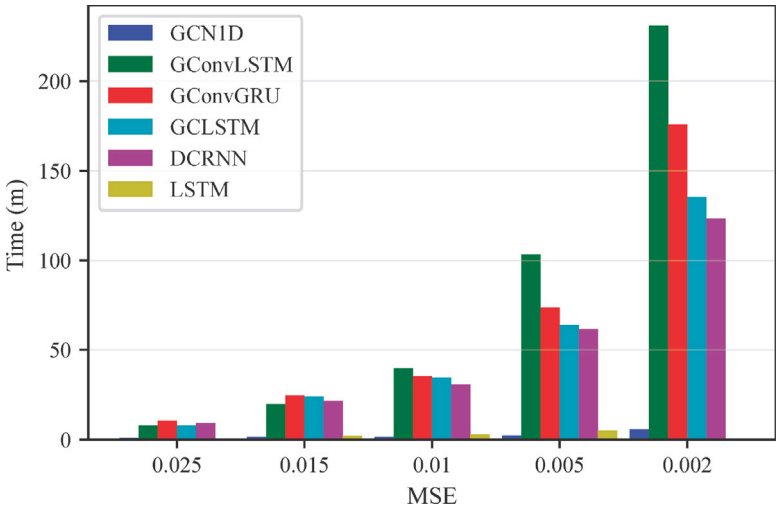


Fig. 6. Histogram plot showing the time it takes for each model to reach fixed level of MSE.

Table 3
Forecasting in Different Seasons.

Season	Night values	MSE	MAE	R^2	Time
Jan-Mar	Yes	43	846	0.993	47.8
Apr-Jun	Yes	64	1033	0.995	47.8
Jul-Set	Yes	55	953	0.996	47.3
Oct-Dec	Yes	78	1034	0.991	46.9
Year	Yes	192	1874	0.980	191.4
Jan-Mar	No	279	3583	0.971	16.5
Apr-Jun	No	180	3024	0.982	24.5
Jul-Set	No	218	3127	0.978	23.5
Oct-Dec	No	299	3831	0.974	16.2
Year	No	299	4013	0.970	81.2

Table 4
Forecasting with Different Prediction Window Sizes.

Input	Output	MSE	MAE	R^2
8	24	1468	6539	0.772
	32	1636	7179	0.741
	48	1369	6491	0.785
	60	1862	7750	0.703
	72	1489	6841	0.763
12	24	464	3435	0.927
	32	490	3721	0.923
	48	546	3979	0.912
	60	799	4913	0.875
	72	532	4046	0.916
24	24	113	1496	0.982
	32	134	1728	0.979
	48	147	1983	0.977
	60	146	2139	0.977
	72	276	2920	0.955
32	24	50	1281	0.992
	32	85	1440	0.987
	48	65	1352	0.990
	60	196	2340	0.969
	72	209	2713	0.967
48	24	20	790	0.997
	32	59	1290	0.991
	48	82	1550	0.987
	60	172	2348	0.973
	72	117	1985	0.981

The solar power production profile is season-dependent. The dataset with which these experiments were conducted concerns a sampling period of 3 months during the winter season. In this work, we have also analyzed how predictions change when trained on different sampling periods, testing the system with different seasons and also with time series collected during the whole year. The numerical results pertaining to these experiments are summarized in the upper half of Table 3.

Since the total number of light hours during a day changes season by season, and this can influence results, we have also tested the system using light-hours values only, so as to outflank fake nocturnal hours predictions and to concentrate only on effective power generation. This solution assumes exact information about sunrise and sunset timing, which is easy to find in a real scenario. We can observe graphical results in Fig. 7. The task is more difficult to perform, but the model is able to predict the profile of the generated power, however with an overall error greater with respect to the classical setup; the numerical results are shown in the lower half of Table 3.

It is worth noting that, when considering all the hours of the day, winter and summer predictions have smaller errors. The results obtained with winter predictions can be caused by the reduced number of light hours and more predictable nocturnal ones, while for the summer predictions the cause may be the low presence of severe atmospheric events. Autumn and spring instead show a greater error most likely for the greater probability of unstable weather. By considering only day values, there is no more component of nocturnal hours which is easier to predict, so the system can be a more validated estimator of the production of solar energy: autumn and winter seasons in fact get a higher error for the reasons set out above.

Moreover, we have analyzed how the model predicts future output power times series varying the length of input and prediction window: we sweep over 5 different input window sizes of 8, 12, 24, 32, and 48 h and for each of them we predicted 5 different prediction windows of 24, 32, 48, 60, and 72 h. The results are shown in Table 4. Parameters and training time do not vary considerably from one test to another, for this reason they are not shown in this table. We remark that, in order to perform a fair analysis for experiments in Tables 3 and 4, we have run a reproducible training for 200 epochs by setting first the random generator's seed of the system.

We can see how a short size of the input sequence does not allow the model to generate accurate predictions. Increasing the input size the model learns to generalize and generate better predictions also for a longer prediction window's size; then, increasing the prediction size logically increases the difficulty of the prediction task.

5.2. Explainability results

In the following, we show the explainability results of the previous forecasting task applied to multivariate time series with several PV

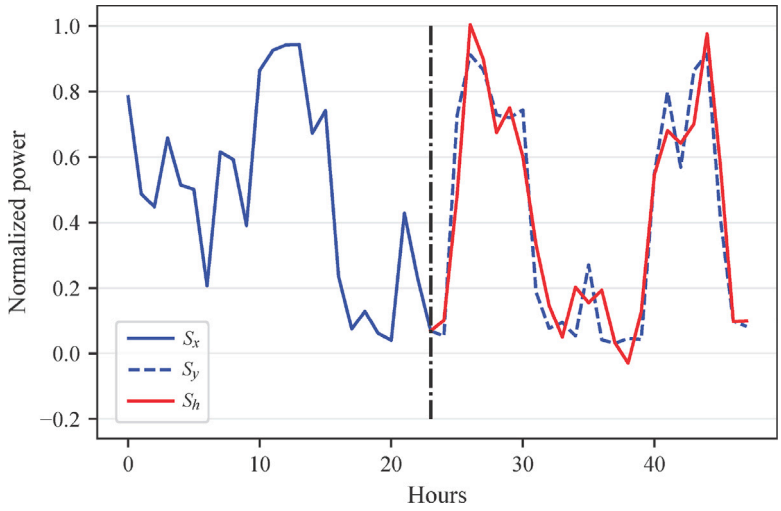


Fig. 7. 1-Day ahead forecasting result for a single node without using time series with values collected during nocturnal hours. The model is fed by the past 24 diurnal hours (S_x) and it generates the prediction (S_h) for the next 24 diurnal hours, which is compared to the ground truth (S_y).

plants. All the explanations are developed using the predictions generated from the proposed GCN1D model. Through the GNNExplainer method, we want to discover the importance map that the elements of the forecasting framework play in the process. First, we have found that local explainability methods applied are suitable to detect important global properties of the energy network and the predictive model.

We discover how the distribution and the distances between nodes in the energy networks play a key role in the forecasting process. Then, we show the outcome of the explainability analysis at a further level of detail, demonstrating the importance of this approach that allows us to visualize and understand what are the most influential and important elements for each node. This way, we can discover what link connection has influenced the most a particular prediction, we can analyze the overall contribution of features or also find out events in past time steps that have influenced the most a node forecast.

5.2.1. Global explanations

These experiments show the GNNExplainer results obtained by iterating the local explainability process for 50 different samples, with respect to the same target node, and collecting the explanations values. These results are summarized in Fig. 8: the darker the node or edge color, the more significant is the element. The target node is labeled with a red circle. Past information of the target node is obviously the most important for future predictions, however for this analysis we have concentrated our attention on other nodes, so explainability values are considered except for the target nodes. Iterating this procedure at different time steps allows us to draw up an overall analysis of the energy network. We can generate assumptions about learned forecasting procedures that can also help system designers during the design and installation process of PV panels.

By construction, the network was built considering only closer nodes to generate edges. From these experiments, we can see how, given a target node, the most important nodes returned by GNNExplainer are neither the closest nor the most distant nodes, but they are those that are at an intermediate distance: the meteorological state is in constant motion, therefore the mid-range nodes are evaluated as the optimal source of information for generating predictions. Subsequently, nearby nodes seem to take more part of the prediction than those very far away; however, this may be a fact that can change for different energy grids. For this reason, the versatility and ease of use of this type of analysis make it very suitable for use in this particular application context.

5.2.2. Local explanations

The outcome of the explainability procedure are the two training masks: the first is applied to the features of each node and the second to the edges of the network, as represented in Figs. 9 and 10, respectively. The node features mask has dimension $m_f \in \mathbb{R}^{N\times(c-2)\times D}$ (we do not mask hours and month information); each value is therefore a measure of the importance of each past time step for a particular feature. These values, respectively for power, temperature and wind variables and for each past time step, are represented in Fig. 9 side by side for visualization.

Observing the image, we can see how the GNNExplainer can carry on the explainability analysis also into the temporal domain: the matrix is sparse, and we can clearly distinguish important elements. In the image, the target node is the number 29, and we find obviously the most important values in the corresponding row. But also we can distinguish that other nodes' values at past time steps are relevant for the prediction of node 29.

We note that the edge mask in Fig. 10 is not the original training mask, since links between nodes are directed edges; we sum the upper and lower triangular parts of the matrix and then we divide by 2 to highlight the effective contribution of a specific link in the transmission of the message. We can observe the degree of importance of each link with respect to the target node for a prediction. The edges with higher values are those that are connected to the target node, but among these, we can discriminate which is the most important, in this case, the number 18.

By combining values for specific dimensions we can generate useful insights for the problem. Results are shown in Fig. 11. We can sum up values with respect to the time dimension and get the most important past time steps for the prediction, which correspond mainly with the diurnal hours. Summing up with respect to node dimension, we can observe which nodes, represented by their features, are the most important. This information can be used together with the edge mask to have a clear view of the importance map of the energy network. Finally, we can also observe which features help the most for the final result: the temperature turns out to be the most important data, even more than the output power. The wind is the least important feature in the forecasting task, but it brings its contribution.

We can also analyze the contribution of features in function of past time steps by means of Fig. 12. Higher values of power production as well as temperature values are the most important. The wind values, although they are less influential in the prediction than temperature and power, seem to have a slight increase of relevance in the forecast

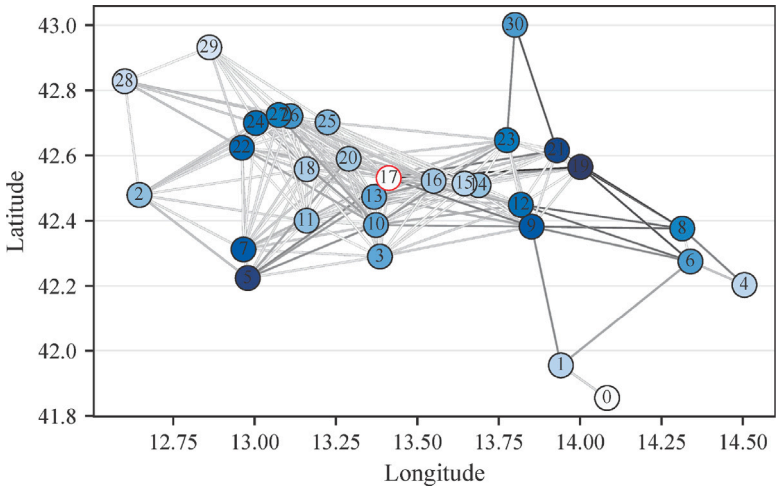


Fig. 8. Final node features and edge importance values reported on the real graph. These global results are obtained by combining explainability values from multiple samples; it shows the overall exchange of relevant information in the network. The more the nodes and links are important to the target node (17), the darker they are represented.

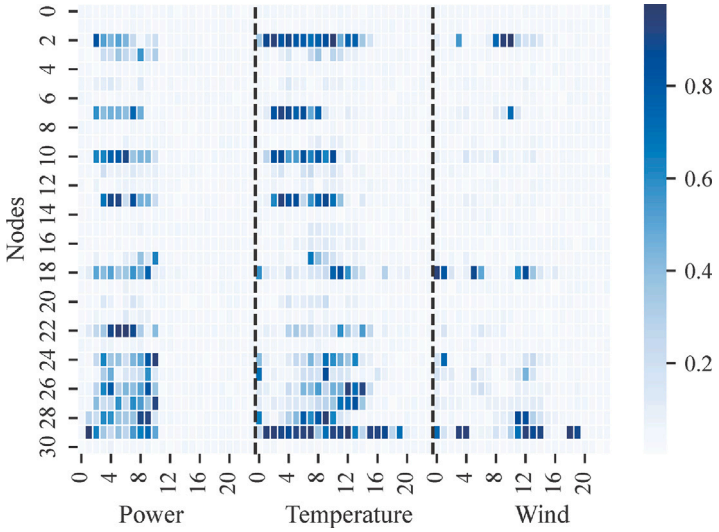


Fig. 9. Node features mask values returned by the GNNExplainer at the end of the training process. Timing information, like months and hours, is not masked.

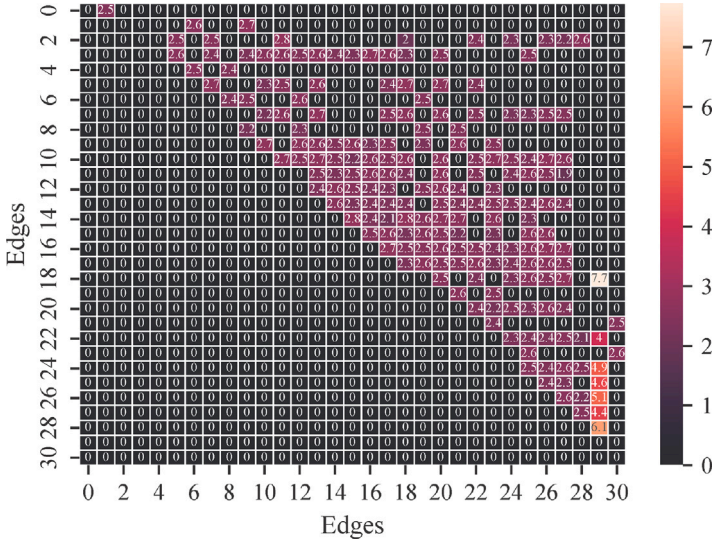


Fig. 10. Trainable edge mask values returned by the GNNExplainer at the end of the training process. Values are referred to the undirected edges that link two nodes.

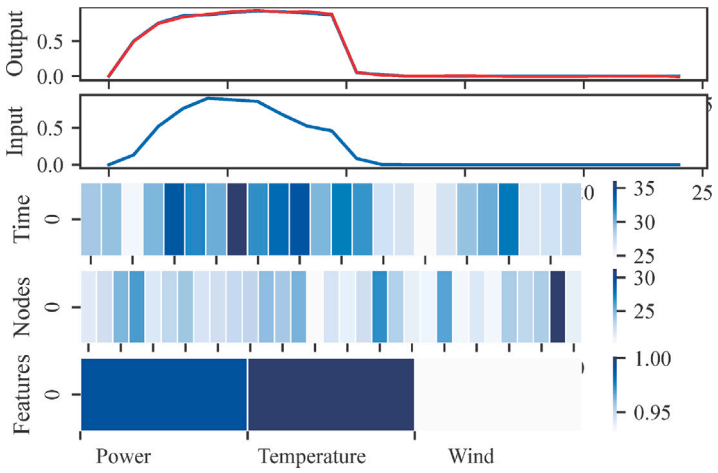


Fig. 11. Given the prediction generated by the model and the input data, both related to the target node, the importance of previous time steps, nodes and features can be visualized by aggregating them and using the values of the training mask. Results of the local explainability method can be handled and the generated explanation at a different level of analysis is examined.

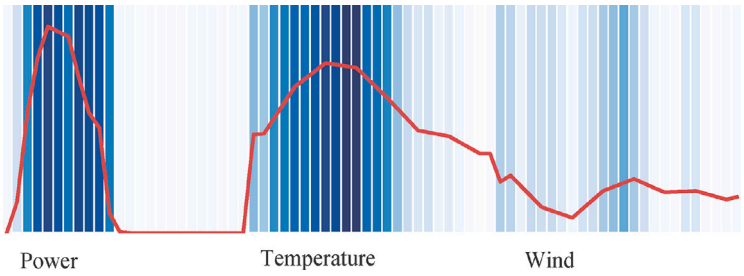


Fig. 12. Importance values of past time steps for each feature: we can analyze singularly each time step and weigh its contribution to the prediction.

in the moments in which the wind speed changes, perhaps to signal the variation of the atmospheric conditions. By using a multi-level analysis of this type can be an optimal tool to discriminate the introduction of new features in the forecasting process. In a real scenario, for each feature coming from a plant we need a specific sensor and this comes at a cost. With this information at hand, we can generate a strategy for an energy network management system.

6. Conclusion

Our research has demonstrated a novel solution to the multi-site and multi-step PV power forecasting task by leveraging the power of STGNNs to integrate prediction and explainability within a single, cohesive framework which provide accurate and transparent forecasts. We have proven how our GCN1D method is a fast and efficient alternative with respect to other STGNN methods. To the best of our knowledge, we are the first that focused on the problem of explainability when using STGNN in power plant forecasting. By applying the GNNExplainer, we were able to generate inferences about the predictions obtained from our model, generating explanations of great relevance for the task. These explanations can be useful also in the creation phase of a new energy network, in order to maximize the production efficiency as a function of real installation costs.

This research was designed with the aim of addressing real-world needs and, as a result, training speed and flexibility were considered as performance evaluation metrics. Our findings can have real practical relevance beyond the benchmarks we consider: countries seeking to optimize their energy production and distribution strategies stand to benefit from the insights provided by our model. Energy distributors and retailers can leverage the explainable predictions generated by our framework to enhance their decision-making processes, leading to a more efficient way to allocate energy and potentially reduce operational costs. In an era where sustainable and cost-efficient energy production is crucial, our research offers a valuable tool to assist and sustain the future of energy systems.

Further possible investigations may concern the development of ad-hoc explainability methodologies for the aforementioned task. New techniques for the construction of the graph can be incorporated into the framework: the selection of the threshold can be learned in order to generate for different energy networks the best graph structure. Or even other relations beyond the simple Euclidean distance, such as the similarity between the various types of construction or installation of PV systems, can be inserted into the graph construction process, relations that explainability methods can help us discover and validate.

CRediT authorship contribution statement

Alessio Verdone: Data curation, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Simone Scardapane:** Conceptualization, Formal analysis, Software, Validation, Visualization. **Massimo Panella:** Conceptualization, Data curation, Methodology, Supervision, Visualization, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Alessio Verdone reports financial support was provided by Government of Italy Ministry of Education University and Research.

Data availability

Data will be made available on request.

Acknowledgments

The contribution of A. Verdone in this work was supported in part by the Italian Ministry of University and Research (MUR), which funded his PhD grant in Information and Communication Technology (ICT) as per the Ministerial Decree no. 1061/2021.

References

[1] Papadis E, Tsatsaronis G. Challenges in the decarbonization of the energy sector. Energy 2020;205:118025.
[2] Wójcik-Jurkiewicz M, Czarnecka M, Kinelski G, Sadowska B, Bilińska-Reformat K. Determinants of decarbonisation in the transformation of the energy sector: The case of Poland. Energies 2021;14(5):1217.
[3] Gielen D, Boshell F, Saygin D, Bazilian MD, Wagner N, Gorini R. The role of renewable energy in the global energy transformation. Energy Strategy Rev 2019;24:38–50.
[4] Ridha E, Nolting L, Praktiknjo A. Complexity profiles: A large-scale review of energy system models in terms of complexity. Energy Strategy Rev 2020;30:100515.
[5] Franco A, Salza P. Strategies for optimal penetration of intermittent renewables in complex energy systems based on techno-operational objectives. Renewable Energy 2011;36(2):743–53.
[6] García Vera YE, Dufo-López R, Bernal-Agustín JL. Energy management in microgrids with renewable energy sources: A literature review. Appl Sci 2019;9(18):3854.
[7] Koohi-Fayegh S, Rosen M. A review of energy storage types, applications and recent developments. J Energy Storage 2020;27:101047.
[8] Giorgi MGD, Congedo PM, Malvoni M. Photovoltaic power forecasting using statistical methods: impact of weather data. IET Sci Meas Technol 2014;8:90–7.
[9] Amarasinghe PAGM, Abeygunawardana NS, Jayasekara TN, Edirisinghe EAJP, Abeygunawardane SK. Ensemble models for solar power forecasting—a weather classification approach. AIMS Energy 2020;8(2):252–71.
[10] Barbieri F, Rajakaruna S, Ghosh A. Very short-term photovoltaic power forecasting with cloud modeling: A review. Renew Sustain Energy Rev 2017;75:242–63.
[11] Rajagukguk RA, Ramadhan RA, Lee H-J. A review on deep learning models for forecasting time series data of solar irradiance and photovoltaic power. Energies 2020;13(24):6623.
[12] Fara L, Diaconu A, Craciunescu D, Fara S. Forecasting of energy production for photovoltaic systems based on ARIMA and ANN advanced models. Int J Photoenergy 2021.
[13] Wang F, Xuan Z, Zhen Z, Li K, Wang T, Shi M. A day-ahead PV power forecasting method based on LSTM-RNN model and time correlation modification under partial daily pattern prediction framework. Energy Convers Manage 2020;212:112766.
[14] Abdel-Nasser M, Mahmoud K. Accurate photovoltaic power forecasting models using deep LSTM-RNN. Neural Comput Appl 2019;31(7):2727–40.
[15] Fernandez-Jimenez LA, Muñoz-Jimenez A, Falces A, Mendoza-Villena M, Garcia-Garrido E, Lara-Santillan PM, et al. Short-term power forecasting system for photovoltaic plants. Renew Energy 2012;44:311–7.
[16] Wang K, Qi X, Liu H. A comparison of day-ahead photovoltaic power forecasting models based on deep learning neural network. Appl Energy 2019;251:113315.
[17] Simeunović J, Schubnel B, Alet P-J, Carrillo RE. Spatio-temporal graph neural networks for multi-site PV power forecasting. IEEE Trans Sustain Energy 2021;13(2):1210–20.
[18] Karimi AM, Wu Y, Koyuturk M, French RH. Spatiotemporal graph neural network for performance prediction of photovoltaic power systems. In: Proceedings of the AAAI conference on artificial intelligence, vol. 35, no. 17. 2021, p. 15323–30.
[19] Khodayar M, Wang J. Spatio-temporal graph deep neural network for short-term wind speed forecasting. IEEE Trans Sustain Energy 2018;10(2):670–81.
[20] Ras G, Xie N, van Gerven M, Doran D. Explainable deep learning: A field guide for the uninitiated. J Artif Intell Res 2022;73:329–96.
[21] Verdone A, Scardapane S, Panella M. Multi-site forecasting of energy time series with spatio-temporal graph neural networks. In: 2022 international joint conference on neural networks. 2022, p. 1–8.
[22] Ying R, Bourgeois D, You J, Zitnik M, Leskovec J. GNNExplainer: Generating explanations for graph neural networks. Adv Neural Inf Process Syst 2019;32:9240–51.
[23] Torres JF, Hadjout D, Sebaa A, Martínez-Álvarez F, Troncoso A. Deep learning for time series forecasting: a survey. Big Data 2021;9(1):3–21.
[24] Dama F, Sinoquet C. Time series analysis and modeling to forecast: a survey. 2021, arXiv:2104.00164.
[25] Bontempi G, Ben Taieb S, Borgne Y-AL. Machine learning strategies for time series forecasting. In: European business intelligence summer school. Springer; 2012, p. 62–77.
[26] Sezer OB, Gudelek MU, Ozbayoglu AM. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. Appl Soft Comput 2020;90:106181.
[27] Li X, Law R, Xie G, Wang S. Review of tourism forecasting research with internet data. Tour Manag 2021;83:104245.
[28] Nunnari G, Nunnari V. Forecasting monthly sales retail time series: a case study. In: 2017 IEEE 19th conference on business informatics, vol. 1. IEEE; 2017, p. 1–6.
[29] Talkhi N, Fatemi NA, Ataei Z, Nooghabi MJ. Modeling and forecasting number of confirmed and death caused COVID-19 in IRAN: A comparison of time series forecasting methods. Biomed Signal Process Control 2021;66:102494.
[30] Chou J-S, Tran D-S. Forecasting energy consumption time series using machine learning techniques based on usage patterns of residential householders. Energy 2018;165:709–26.
[31] Das UK, Tey KS, Seyedmahmoudian M, Mekhilef S, Idris MYI, Van Deventer W, et al. Forecasting of photovoltaic power generation and model optimization: A review. Renew Sustain Energy Rev 2018;81:912–28.
[32] Larson DP, Nonnenmacher L, Coimbra CF. Day-ahead forecasting of solar power output from photovoltaic plants in the American Southwest. Renew Energy 2016;91:11–20.
[33] Wu Y-K, Chen C-R, Abdul Rahman H. A novel hybrid model for short-term forecasting in PV power generation. Int J Photoenergy 2014;2014:1–9.
[34] Gigoni L, Betti A, Crisostomi E, Franco A, Tucci M, Bizzarri F, et al. Day-ahead hourly forecasting of power generation from photovoltaic plants. IEEE Trans Sustain Energy 2018;9(2):831–42.
[35] Rosato A, Panella M, Andreotti A, Mohammed OA, Araneo R. Two-stage dynamic management in energy communities using a decision system based on elastic net regularization. Appl Energy 2021;291:116852.
[36] Severiano CA, Silva PCL, Sadaei HJ, Guimarães FG. Very short-term solar forecasting using fuzzy time series. In: 2017 IEEE international conference on fuzzy systems. 2017, p. 1–6.
[37] Azadeh A, Tarverdian S. Integration of genetic algorithm, computer simulation and design of experiments for forecasting electrical energy consumption. Energy Policy 2007;35(10):5229–41.
[38] Succetti F, Rosato A, Panella M. An adaptive embedding procedure for time series forecasting with deep neural networks. Neural Netw 2023;167:715–29.
[39] Kiranyaz S, Ince T, Abdeljaber O, Avci O, Gabbouj M. 1-D convolutional neural networks for signal processing applications. In: ICASSP 2019 - 2019 IEEE international conference on acoustics, speech and signal processing. 2019, p. 8360–4.
[40] Liang S, Nguyen LH, Jin F. A multi-variable stacked long-short term memory network for wind speed forecasting. In: 2018 IEEE international conference on big data. 2018, p. 4561–4.
[41] Hossain MS, Mahmood H. Short-term photovoltaic power forecasting using an LSTM neural network and synthetic weather forecast. IEEE Access 2020;8:172524–33.
[42] Succetti F, Rosato A, Araneo R, Panella M. Deep neural networks for multivariate prediction of photovoltaic power time series. IEEE Access 2020;8:211490–505.
[43] Oreshkin BN, Carpov D, Chapados N, Bengio Y. N-BEATS: neural basis expansion analysis for interpretable time series forecasting. In: 8th international conference on learning representations. 2020, p. 1–31.
[44] Gao C, Zhang N, Li Y, Bian F, Wan H. Self-attention-based time-variant neural networks for multi-step time series forecasting. Neural Comput Appl 2022;34.
[45] Grigsby J, Wang Z, Nguyen N, Qi Y. Long-range transformers for dynamic spatiotemporal forecasting. 2023, arXiv:2109.12218.
[46] Zhou T, Ma Z, Wen Q, Wang X, Sun L, Jin R. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In: Chaudhuri K, Jegelka S, Song L, Szepesvari C, Niu G, Sabato S, editors. Proceedings of the 39th international conference on machine learning. Proceedings of machine learning research, vol. 162, PMLR; 2022, p. 27268–86.
[47] Zhou H, Zhang S, Peng J, Zhang S, Li J, Xiong H, et al. Informer: Beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI conference on artificial intelligence, vol. 35, no. 12. 2021, p. 11106–15.
[48] Wu H, Xu J, Wang J, Long M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. 2022, arXiv:2106.13008.
[49] Mohamed A, Qian K, Elhoseiny M, Claudel C. Social-STGCNN: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In: 2020 IEEE/CVF conference on computer vision and pattern recognition. 2020, p. 14412–20.
[50] Kan X, Cui H, Lukemire J, Guo Y, Yang C. FBNETGEN: Task-aware GNN-based fMRI analysis via functional brain network generation. In: Konukoglu E, Menze B, Venkataraman A, Baumgartner C, Dou Q, Albarqouni S, editors. Proceedings of the 5th international conference on medical imaging with deep learning. Proceedings of machine learning research, vol. 172, PMLR; 2022, p. 618–37.
[51] Jiang W, Luo J. Graph neural network for traffic forecasting: A survey. Expert Syst Appl 2022;207:117921.
[52] Zhao L, Song Y, Zhang C, Liu Y, Wang P, Lin T, et al. T-GCN: A temporal graph convolutional network for traffic prediction. IEEE Trans Intell Transp Syst 2020;21(9):3848–58.
[53] Yu B, Yin H, Zhu Z. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In: Proceedings of the twenty-seventh international joint conference on artificial intelligence. International Joint Conferences on Artificial Intelligence Organization; 2018.

[54] Panagopoulos G, Nikolentzos G, Vazirgiannis M. Transfer graph neural networks for pandemic forecasting. In: The thirty-fifth AAAI conference on artificial intelligence. 2021, p. 4838–45.
[55] Fang Z, Long Q, Song G, Xie K. Spatial-temporal graph ODE networks for traffic flow forecasting. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining. New York, NY, USA: Association for Computing Machinery; 2021, p. 364–73.
[56] Zheng C, Fan X, Wang C, Qi J. GMAN: a graph multi-attention network for traffic prediction. In: The thirty-fourth AAAI conference on artificial intelligence, AAAI 2020, the thirty-second innovative applications of artificial intelligence conference, IAAI 2020, the tenth AAAI symposium on educational advances in artificial intelligence. AAAI Press; 2020, p. 1234–41.
[57] Qin Y, Zhao F, Fang Y, Luo H, Wang C. Memory attention enhanced graph convolution long short-term memory network for traffic forecasting. Int J Intell Syst 2022;37(9):6555–76.
[58] Bai L, Yao L, Li C, Wang X, Wang C. Adaptive graph convolutional recurrent network for traffic forecasting. In: Proceedings of the 34th international conference on neural information processing systems. Red Hook, NY, USA: Curran Associates Inc.; 2020, p. 1–12.
[59] Seo Y, Defferrard M, Vandergheynst P, Bresson X. Structured sequence modeling with graph convolutional recurrent networks. In: Cheng L, Leung ACS, Ozawa S, editors. Neural information processing. Cham: Springer International Publishing; 2018, p. 362–73.
[60] Chen J, Wang X, Xu X. GC-LSTM: graph convolution embedded LSTM for dynamic network link prediction. Appl Intell 2022;52:7513–28.
[61] Li Y, Yu R, Shahabi C, Liu Y. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. 2018, arXiv:1707.01926.
[62] Danilevsky M, Qian K, Aharonov R, Katsis Y, Kawas B, Sen P. A survey of the state of explainable AI for natural language processing. In: Proceedings of the 1st conference of the asia-pacific chapter of the association for computational linguistics and the 10th international joint conference on natural language processing. Suzhou, China: Association for Computational Linguistics; 2020, p. 447–59.
[63] Zhang X, Chan FT, Mahadevan S. Explainable machine learning in image classification models: An uncertainty quantification perspective. Knowl-Based Syst 2022;243:108418.
[64] Joshi G, Walambe R, Kotecha K. A review on explainability in multimodal deep neural nets. IEEE Access 2021;9:59800–21.
[65] Amann J, Blasimme A, Vayena E, Frey D, Madai V. Explainability for artificial intelligence in healthcare: a multidisciplinary perspective. BMC Med Inform Decis Mak 2020;20.
[66] Freeborough W, van Zyl T. Investigating explainability methods in recurrent neural network architectures for financial time series data. Appl Sci 2022;12(3):1427.
[67] Yuan H, Yu H, Gui S, Ji S. Explainability in graph neural networks: A taxonomic survey. IEEE Trans Pattern Anal Mach Intell 2023;45(05):5782–99.
[68] Baldassarre F, Azizpour H. Explainability techniques for graph convolutional networks. 2019, arXiv:1905.13686.
[69] Pope PE, Kolouri S, Rostami M, Martin CE, Hoffmann H. Explainability methods for graph convolutional neural networks. In: 2019 IEEE/CVF conference on computer vision and pattern recognition. 2019, p. 10764–73.
[70] Luo D, Cheng W, Xu D, Yu W, Zong B, Chen H, et al. Parameterized explainer for graph neural network. In: Proceedings of the 34th international conference on neural information processing systems. Red Hook, NY, USA: Curran Associates Inc.; 2020, p. 1–12.
[71] Funke T, Khosla M, Anand A. Hard masking for explaining graph neural networks. 2021, URL https://openreview.net/forum?id=uDN8pRAdsoC.
[72] Schlichtkrull MS, Cao ND, Titov I. Interpreting graph neural networks for NLP with differentiable edge masking. 2022, arXiv:2010.00577.
[73] Huang Q, Yamada M, Tian Y, Singh D, Chang Y. GraphLIME: Local interpretable model explanations for graph neural networks. IEEE Trans Knowl Data Eng 2023;35(7):6968–72.
[74] Vu MN, Thai MT. PGM-Explainer: Probabilistic Graphical Model Explanations for Graph Neural Networks. In: Proceedings of the 34th international conference on neural information processing systems. Red Hook, NY, USA: Curran Associates Inc.; 2020, p. 1–11.
[75] Zhang Y, Defazio D, Ramesh A. RelEx: A model-agnostic relational model explainer. In: Proceedings of the 2021 AAAI/ACM conference on AI, ethics, and society. New York, NY, USA: Association for Computing Machinery; 2021, p. 1042–9.
[76] Schnake T, Eberle O, Lederer J, Nakajima S, Schutt KT, Mueller K-R, et al. Higher-order explanations of graph neural networks via relevant walks. IEEE Trans Pattern Anal Mach Intell 2021;1.
[77] Yuan H, Tang J, Hu X, Ji S. Xgnn: Towards model-level explanations of graph neural networks. In: Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. 2020, p. 430–8.
[78] Spinelli I, Scardapane S, Uncini A. Missing data imputation with adversarially-trained graph convolutional networks. Neural Netw 2020;129:249–60.
[79] Kazi A, Cosmo L, Ahmadi S-A, Navab N, Bronstein MM. Differentiable graph module (DGM) for graph convolutional networks. IEEE Trans Pattern Anal Mach Intell 2023;45(2):1606–17.
[80] Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R, editors. Advances in neural information processing systems, vol. 30. Curran Associates, Inc.; 2017, p. 1–11.
[81] Bruna J, Zaremba W, Szlam A, LeCun Y. Spectral networks and locally connected networks on graphs. In: Bengio Y, LeCun Y, editors. 2nd international conference on learning representations, ICLR 2014, Banff, AB, Canada, April 14–16, 2014, conference track proceedings. 2014, p. 1–14.
[82] Defferrard M, Bresson X, Vandergheynst P. Convolutional neural networks on graphs with fast localized spectral filtering. In: Proceedings of the 30th international conference on neural information processing systems. Red Hook, NY, USA: Curran Associates Inc.; 2016, p. 3844–52.
[83] Kiranyaz S, Avci O, Abdeljaber O, Ince T, Gabbouj M, Inman DJ. 1D convolutional neural networks and applications: A survey. Mech Syst Signal Process 2021;151:107398.
[84] Dabkowski P, Gal Y. Real time image saliency for black box classifiers. In: Proceedings of the 31st international conference on neural information processing systems. Red Hook, NY, USA: Curran Associates Inc.; 2017, p. 6970–9.
[85] Huld T, Müller R, Gambardella A. A new solar radiation database for estimating PV performance in Europe and Africa. Sol Energy 2012;86(6):1803–15.
[86] Fey M, Lenssen JE. Fast graph representation learning with PyTorch Geometric. In: ICLR 2019 workshop on representation learning on graphs and manifolds. 2019, p. 1–9.
[87] Rozemberczki B, Scherer P, He Y, Panagopoulos G, Riedel A, Astefanoaei M, et al. Pytorch geometric temporal: Spatiotemporal signal processing with neural machine learning models. In: Proceedings of the 30th ACM international conference on information & knowledge management. New York, NY, USA: Association for Computing Machinery; 2021, p. 4564–73.
[88] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 1997;9:1735–80.
[89] Kingma DP, Ba J. Adam: A method for stochastic optimization. In: Bengio Y, LeCun Y, editors. 3rd international conference on learning representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, conference track proceedings. 2015, p. 1–13.