This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3040042, IEEE Transactions on Signal and Information Processing over Networks

Joint Forecasting and Interpolation of Time-Varying Graph Signals Using Deep Learning

Gabriela Lewenfus, Wallace A. Martins, *Senior Member, IEEE,* Symeon Chatzinotas, *Senior Member, IEEE,* Björn Ottersten, *Fellow, IEEE*

***Abstract*—We tackle the problem of forecasting network-signal snapshots using past signal measurements acquired by a subset of network nodes. This task can be seen as a combination of multivariate time-series forecasting (temporal prediction) and graph-signal interpolation (spatial prediction). This is a fundamental problem for many applications wherein deploying a high granularity network is impractical. Our solution combines recurrent neural networks with frequency-analysis tools from graph signal processing, and assumes that data is sufficiently smooth with respect to the underlying graph. The proposed learning model outperforms state-of-the-art deep learning techniques, especially when predictions are made using a small subset of network nodes, considering two distinct real world datasets: temperatures in the US and speed flow in Seattle. The results also indicate that our method can handle noisy signals and missing data, making it suitable to many practical applications.**

***Index Terms*—Multivariate time series, forecasting and interpolation, deep learning, recurrent neural networks (RNNs), graph signal processing (GSP)**

I. Introduction

SPATIOTEMPORAL (ST) prediction is a fundamental abstract problem featuring in many practical applications, including climate analyses [1], transportation management [2], neuroscience [3], electricity markets [4], and several geographical phenomenon analyses [5]. The temperature in a city, for instance, is influenced by its location, by the season, and even by the hour of the day. Another example of data with ST dependencies is the traffic state of a road, since it is influenced by adjacent roads and also by the hour of the day. ST prediction boils down to *forecasting* (temporal prediction) and *interpolation* (spatial prediction). The former refers to predicting some physical phenomenon using historical data acquired by a network of spatially-distributed sensors. The latter refers to predicting the phenomenon with a higher spatial resolution. In this context, ST data can be seen as a network signal in which a time series is associated with each network element; the dynamics (time-domain evolution) of the time series depends on the network structure (spatial domain), rather than on the isolated network elements only. The interpolation is useful to generate a denser (virtual) network.

Classical predictive models generally assume independence of data samples and disregard relevant spatial information [6], [7]. Vector autoregressive (VAR) [8], a statistical multivariate model, and machine learning (ML) approaches, such as support vector regression (SVR) [9] and random forest regression [10], can achieve higher accuracy than classical predictive models; yet, they fail to fully capture spatial relations. More recently, some progress has been made by applying neural networks[1] (NNs) to predict ST data [1], [2], [11]–[14]. NNs have the capacity of not only mapping an input data to an output, but also of learning a useful representation to improve the mapping accuracy [15]. Nonetheless, due to their high complexity, the fully-connected architectures of these NNs may fail to extract simultaneous spatial and temporal features from data, making it difficult to generalize the model.

In order to learn spatial information from these multivariate time series, some works have combined convolutional NNs (CNNs) with recurrent NNs (RNNs), such as long short-term memory (LSTM) [16]–[20]. However, CNNs are restricted to grid-like uniformly structured data, such as images and videos. To overcome this issue, and inspired by graph[2] signal processing (GSP), some works have developed convolutions on graph-structured data (graph signals) [21]–[25], which have been used in combination with either RNN, time convolution, and/or attention mechanisms to make predictions in a variety of applications. These works are summarized in TABLE I.

GSP theory has been applied to analyze/process many irregularly structured datasets in several applications [62]. An import task addressed by GSP is interpolation on graphs, i.e., (spatially) predicting the signals on a subset of graph nodes based on known signal values from other nodes [63]. In general, graph interpolation is based on local or global approaches. Local methods, such as k-nearest neighbors (k-NNs) [64], compute the unknown signal values in a set of network nodes using values from their closest neighbors, being computationally efficient. Global methods, on the other hand, interpolate the unknown signal values at once and can provide better results by taking the entire network into account at the expensive of a higher computational burden [63], [65]. Many GSP interpolation schemes have been proposed [66]–[70].

It is not always possible to deploy a very large number of sensors due to limited physical space or budget constraints; for example, placing many electrodes at once in the human

Ms. Lewenfus is with the Federal University of Rio de Janeiro (UFRJ), (e-mail: gabriela.lewenfus@gmail.com); Prof. Martins is with UFRJ (on leave) and with the University of Luxembourg (as Research Associate), (e-mail: wallace.martins@smt.ufrj.br); Prof. Chatzinotas is with the University of Luxembourg (SnT), (e-mail: symeon.chatzinotas@uni.lu); and Prof. Ottersten is with the University of Luxembourg (SnT), (e-mail: bjorn.ottersten@uni.lu)

This work was financed in part by the ERC project AGNOSTIC (Actively Enhanced Cognition based Framework for Design of Complex Systems – Grant number: 742648) and by the Brazilian agencies CNPq and Faperj.

[1] In this paper the word “network” can refer to a neural network in the context of deep learning or a physical network that is represented by a graph.

[2] Graphs are mathematical structures able to represent rather general datasets, including ST data with irregular domains, as in sensor networks.

TABLE I: Summary of recent works that use deep learning to predict ST data. First column defines the application. Second and third columns refer to the spatial and temporal techniques employed, respectively. "Conv." means temporal convolution; AE means auto encoder; RBM means restricted Boltzman machine; "other" encompasses other predictive strategies, such as attention mechanisms

Application	GSP	Temporal	Reference
traffic	Yes	RNN	[25], [26]–[31]
		Conv.	[32]–[37]
		other	[38], [39]
	No	RNN	[17], [20], [40]–[44],
		other	[2] , [19], [45]–[49]
wind	Yes	RNN	[50]
	No	RNN	[14]
		other	[51]–[53]
meteorological	Yes	AE	[54]
	No	RNN	[16]
		other	[5], [55]
body-motion	Yes	RNN	[56]–[58]
neuroscience	Yes	Conv.	[59]
	No	RBM	[60]
semantic	Yes	RNN	[61]

cortex may be unfeasible. Installation and maintenance costs of devices can also limit the number of sensors deployed in a network [71]. In order to better elucidate this problem, consider a network of sensors and suppose that there is interest in collecting signals over spatial points where there are no deployed sensors. The entire network, composed of deployed and non-deployed sensors, can be represented by a "virtual" graph, in which the deployed sensors comprise a sample of nodes (i.e., they form a sparser graph). Thus, developing a predictive model capable of forecasting (temporal prediction) and interpolating (spatial prediction) time-varying signals defined on graph nodes can be of great applicability. This problem can be regarded as a semi-supervised task, since only part of the nodes is available for training. Other works have addressed this problem: in [72] the graph is extended to incorporate the time dimension and a kernel-based algorithm is used for prediction; this approach, therefore, relies on the assumption of smoothness in the time domain, which is not reasonable for many applications, such as traffic flow prediction. In [50], the ST wind speed model is evaluated in a semi-supervised framework in which only part of the nodes is used for training the model, while interpolation is performed only in test phase. Therefore, the parameters learned during the training phase do not take into account the interpolation aspect. Note that the joint forecasting and interpolation is slightly different from tracking, which is already tackled by the GSP literature [73], [74]; in tracking, the values associated with the in-sample nodes in the current timestep are available and only the values associated with the out-of-sample nodes in the current timestep are targeted by the model, whereas, in prediction, the entire graph signal (GS) in the current timestep is targeted.

Two straightforward solutions to deal with the problem of forecasting and interpolating sampled GSs are:[3] (i) applying a forecasting model to the input GS and then interpolating the output; or (ii) interpolating the sampled GS and then feeding it to the forecasting model. These solutions tackle the ST prediction task separately and may fail to capture the inherent coupling between time and space domains, especially due to the low availability of spatial information. In this paper, a graph-based NN architecture is proposed to handle ST correlations by employing GSP in conjunction with a gated-recurrent unit (GRU). Thus, we address the inherent nature of ST data by jointly forecasting and interpolating the underlying network signals. A global interpolation approach is adopted as it provides accurate results when the signal is smooth in the GSP sense, whereas an RNN forecasting model is adopted given its prior success in network prediction. We consider that both the sampled GS and its spectral components — i.e., the Fourier coefficients, which carry spatial information on the underlying graph — work as inputs to a predictive model. The major contribution of the proposed learning model is, therefore, the ability to predict ST data by observing only a few nodes of the entire network.

Considering the proposed learning model, we introduce four possible classes of problems:

- supervised applications, where the labels of all nodes are available for training but only a fixed subset of graph nodes can be used as input to the model in the test phase;
- semi-supervised application, wherein only data associated with a subset of nodes are available for training and computing gradients;
- noise-corrupted application, in which all nodes are available during the entire process, but additive noise corrupts the network signals;
- missing-value application, where a time-varying fraction of nodes are available for testing, but all nodes can be used for training.

The proposed learning model outperforms (in terms of root mean square error) all tested deep learning (DL) based benchmarks in 27 out of the 30 tested scenarios.

The paper is organized as follows: Section II presents some fundamental aspects of GSP, focusing on the sampling theory that will be used to build the interpolation module of the proposed learning model. Section III describes the new learning framework. Section IV describes four classes of applications that can benefit from the proposal. Section V presents the numerical results and related discussions. Section VI contains the concluding remarks of the paper.

II. Background

Let $\mathcal{G} \triangleq (\mathcal{V}, \mathcal{E}, \mathbf{A})$ be a weighted undirected and connected graph, where $\mathcal{V} \triangleq \{v_1, \ldots, v_N\}$ is the set of N nodes, $\mathcal{E}$ is the set of edges, and $\mathbf{A}$ is the $N \times N$ adjacency matrix containing edge weights A_{mn}. The adjacency matrix can be a similarity

[3] A network signal to be interpolated can be initially modeled by a GS, which in turn can be regarded as a sampled version of a (higher dimension) GS defined over a denser set of nodes belonging to a virtual graph.

TABLE II: Notations

Notation	Definition
$\mathcal{G}$	graph
$\mathcal{V}$	entire set of graph nodes
$\mathcal{S}$	subset of graph nodes
$\mathcal{F}$	subset of graph spectrum
$\overline{\mathcal{S}}$	the complement of the set $\mathcal{S}$
$\mathbf{L}$	Laplacian matrix
$\mathbf{U}$	matrix of Laplacian eigenvectors
$\mathbf{U}_{:,\mathcal{F}}$	submatrix of $\mathbf{U}$ with columns in the set $\mathcal{F}$
$\mathbf{U}_{\mathcal{S},\mathcal{F}}$	submatrix of $\mathbf{U}$ with columns in $\mathcal{F}$ and rows in $\mathcal{S}$
$\boldsymbol{\Psi}_\mathcal{S}$	sampling operator $\mathcal{V} \to \mathcal{S}$
$\boldsymbol{\Phi}_\mathcal{S}$	interpolation operator $\mathcal{S} \to \mathcal{V}$
$\mathbf{x}_\mathcal{S}$	signal $\mathbf{x}$ restricted to the set $\mathcal{S}$ (sampled GS)
$\hat{\mathbf{x}}_\mathcal{F}$	frequency content of the GS $\mathbf{x}$ restricted to $\mathcal{F}$

matrix or be built based on prior information, such as nodes' locations in the physical network. A GS is a real-valued scalar function $x : \mathcal{V} \to \mathbb{R}$ taking values on the graph nodes, and it will be represented by the N-dimensional vector $\mathbf{x}$ with entries $[\mathbf{x}]_n = x_n = x(v_n)$.

The diagonal matrix $\mathbf{D} \in \mathbb{R}^{N\times N}$ is the degree matrix in which $D_{nn} = \sum_m A_{nm}$ measures the connectivity degree of each node.

Most of the graph convolutions in the literature are based on the Laplacian matrix $\mathbf{L} \triangleq \mathbf{D} - \mathbf{A}$, which is a semi-definite matrix, for a symmetric adjacency $\mathbf{A}$. Let $\mathbf{U}$ be the matrix of orthonormal eigenvectors of $\mathbf{L}$. The graph Fourier transform (GFT) of the GS $\mathbf{x}$ is $\hat{\mathbf{x}} \triangleq \mathbf{U}^{\mathrm{T}}\mathbf{x}$ and the eigenvalues of $\mathbf{L}$, $\lambda_1, \dots, \lambda_N \geq 0$, are considered as the graph frequencies. The eigenvectors of the adjacency matrix can also be used as the Fourier basis [75]. For the reader's convenience, TABLE II contains the main notations that will be used in this work.

A. GS Sampling

Let $\mathcal{S} \triangleq \{s_1, ..., s_M\} \subset \mathcal{V}$ be a subset of nodes with $M \leq N$ nodes; the vector of measurements $\mathbf{x}_\mathcal{S} \in \mathbb{R}^M$ is given by $\mathbf{x}_\mathcal{S} = \boldsymbol{\Psi}_\mathcal{S}\mathbf{x}$, where the sampling operator

$$[\boldsymbol{\Psi}_\mathcal{S}]_{mn} = \begin{cases} 1, & \text{if } v_n = s_m \\ 0, & \text{otherwise} \end{cases} \qquad (1)$$

selects from $\mathcal{V}$ the nodes in $\mathcal{S}$. The interpolation operator $\boldsymbol{\Phi}_\mathcal{S}$ is an $N \times M$ matrix such that the recovered signal is $\tilde{\mathbf{x}} = \boldsymbol{\Phi}_\mathcal{S}\boldsymbol{\Psi}_\mathcal{S}\mathbf{x}$. If $\tilde{\mathbf{x}} = \mathbf{x}$, the pair of sampling and interpolation operators $(\boldsymbol{\Phi}_\mathcal{S}, \boldsymbol{\Psi}_\mathcal{S})$ can perfectly recover the signal $\mathbf{x}$ from its sampled version. As the rank of $\boldsymbol{\Phi}_\mathcal{S}\boldsymbol{\Psi}_\mathcal{S}$ is smaller or equal to M, having $\tilde{\mathbf{x}} = \mathbf{x}$ is not possible for all $\mathbf{x} \in \mathbb{R}^N$ when $M < N$. However, perfect reconstruction can be achieved for a class of bandlimited GS.

The GS $\mathbf{x}_\mathrm{b}$ is said $\mathcal{F}$-bandlimited if $[\hat{\mathbf{x}}_\mathrm{b}]_n = 0 \ \forall n$ such that $\lambda_n \notin \mathcal{F} \subset \{\lambda_1, \dots, \lambda_N\}$, that is, the frequency content of $\mathbf{x}_\mathrm{b}$ is restricted to the set of frequencies $\mathcal{F}$. Some works also restrict the support of the frequency content and consider that a GS $\mathbf{x}_\mathrm{b}$ is ω-bandlimited if $[\hat{\mathbf{x}}_\mathrm{b}]_n = 0 \ \forall n$ such that $\lambda_n > \omega$ [76]. In this paper, a bandlimited signal is a sparse vector in the GFT domain. The following theorem guarantees the perfect reconstruction of an $\mathcal{F}$-bandlimited GS for some sampling sets.

Theorem 1 *([67], [66]) If the sampling operator $\boldsymbol{\Psi}_\mathcal{S}$ satisfies*

$$\mathrm{rank}(\boldsymbol{\Psi}_\mathcal{S}\mathbf{U}_{:,\mathcal{F}}) = |\mathcal{F}| = K, \qquad (2)$$

then

$$\mathbf{x}_\mathrm{b} = \boldsymbol{\Phi}_\mathcal{S}\boldsymbol{\Psi}_\mathcal{S}\mathbf{x}_\mathrm{b} \qquad (3)$$

as long as $\boldsymbol{\Phi}_\mathcal{S} = \mathbf{U}_{:,\mathcal{F}}\boldsymbol{\Sigma}$, where $\boldsymbol{\Sigma}$ satisfies $\boldsymbol{\Sigma}\boldsymbol{\Psi}_\mathcal{S}\mathbf{U}_{:,\mathcal{F}} = \mathbf{I}_K$ and $\mathbf{U}_{:,\mathcal{F}}$ is a submatrix of $\mathbf{U}$ with columns restricted to the indices associated with the frequencies in $\mathcal{F}$.

The condition in (2) is also equivalent to

$$\mathrm{SV}_{\max}(\mathbf{U}_{\overline{\mathcal{S}},\mathcal{F}}) \leq 1, \qquad (4)$$

where $\mathrm{SV}_{\max}(.)$ stands for the largest singular value [77] and $\overline{\mathcal{S}} = \mathcal{V} \setminus \mathcal{S}$. This means that no $\mathcal{F}$-bandlimited signal over the graph $\mathcal{G}$ is supported on $\overline{\mathcal{S}}$.

In order to have $\boldsymbol{\Sigma}\boldsymbol{\Psi}_\mathcal{S}\mathbf{U}_{:,\mathcal{F}} = \mathbf{I}_K$ we must have $M \geq K$, since $\mathrm{rank}(\mathbf{U}_{:,\mathcal{F}}) = K$. If $M \geq K$, $\boldsymbol{\Sigma}$ is the pseudo-inverse of $\boldsymbol{\Psi}_\mathcal{S}\mathbf{U}_{:,\mathcal{F}}$ and the interpolation operator is

$$\boldsymbol{\Phi} = \mathbf{U}_{:,\mathcal{F}}(\mathbf{U}^{\mathrm{T}}_{:,\mathcal{F}}\boldsymbol{\Psi}_\mathcal{S}\mathbf{U}_{:,\mathcal{F}})^{-1}\mathbf{U}^{\mathrm{T}}_{\mathcal{S},\mathcal{F}}\,. \qquad (5)$$

Since $\mathbf{U}$ is non-singular, there is always at least a subset $\mathcal{S}$ such that the condition in (2) is satisfied. Nonetheless, for many choices of $\mathcal{S}$, $\boldsymbol{\Psi}_\mathcal{S}\mathbf{U}_{:,\mathcal{F}}$ can be full rank but ill-conditioned, leading to large reconstruction errors, especially in the presence of noisy measurements or in the case of approximately bandlimited GS. To overcome this issue, optimal sampling strategies, in the sense of minimizing reconstruction error, can be employed [67]. Note that $\boldsymbol{\Phi}$ depends on both $\mathcal{S}$ and $\mathcal{F}$, but this dependence is omitted for simplicity's sake.

B. Approximately Bandlimited GS

In practice, most GSs are only approximately bandlimited [78]. A GS is approximately $(\mathcal{F}, \epsilon)$-bandlimited if [77]

$$\mathbf{x} = \mathbf{x}_\mathrm{b} + \boldsymbol{\eta}, \qquad (6)$$

where $\mathbf{x}_\mathrm{b}$ is an $\mathcal{F}$-bandlimited GS and $\boldsymbol{\eta}$ is an $\overline{\mathcal{F}}$-bandlimited GS such that $\|\boldsymbol{\eta}\|_2 < \epsilon$. If signal $\mathbf{x}$ is sampled on the subset $\mathcal{S}$ and recovered by the interpolator in (5), the error energy of the reconstructed signal is upper bounded by

$$\|\tilde{\mathbf{x}} - \mathbf{x}\|_2 \leq \frac{\|\boldsymbol{\eta}\|_2}{\cos(\theta_{\mathcal{S},\mathcal{F}})}, \qquad (7)$$

where $\theta_{\mathcal{S},\mathcal{F}}$ is the maximum angle between the subspace of signals supported on $\mathcal{S}$ and the subspace of $\mathcal{F}$-bandlimited GS. It can be shown that $\cos(\theta_{\mathcal{S},\mathcal{F}}) = \mathrm{SV}_{\min}(\boldsymbol{\Psi}_\mathcal{S}\mathbf{U}_{:,\mathcal{F}})$; therefore, in order to minimize the upper bound of the reconstruction error in (7), the set $\mathcal{S}$ should maximize $\mathrm{SV}_{\min}(\boldsymbol{\Psi}_\mathcal{S}\mathbf{U}_{:,\mathcal{F}})$. Finding the optimal set $\mathcal{S}$ is a combinatorial optimization problem, equivalent to the E-optimal design [79], that can require an exhaustive search in all possible subsets of $\mathcal{V}$ with size M. A suboptimal solution can be obtained by the greedy search in [67, Algorithm 1].

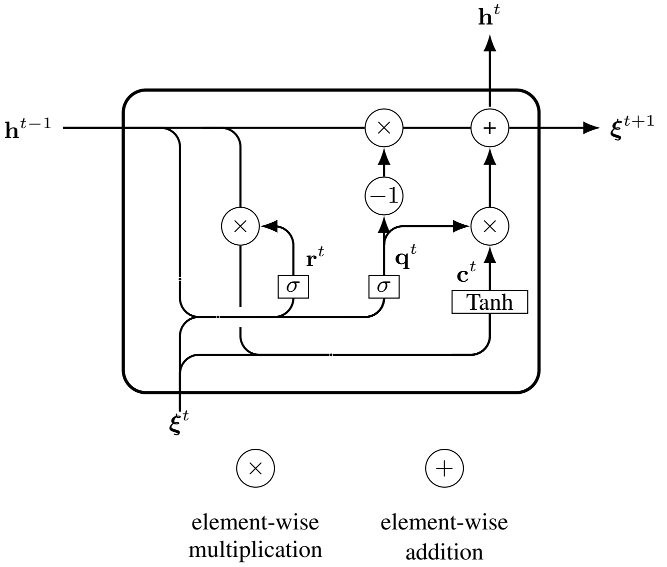


Fig. 1: GRU cell.

C. Gated-Recurrent Unit

The proposed learning model described in Section III employs a GRU cell [80] as the basic building block. The GRU cell is composed of a hidden state $\mathbf{h}^t$, which allows the weights of the GRU to be shared across time, as well as by two gates $\mathbf{q}^t$ and $\mathbf{r}^t$, which modulate the flow of information inside the cell unit. Fig. 1 depicts the architecture of a GRU, where $\boldsymbol{\xi}^t$ and $\boldsymbol{\xi}^{t+1}$ are the input and output signals, respectively. The gates are given by:

$$\mathbf{q}^t = \sigma(\mathbf{W}_q\boldsymbol{\xi}^t + \mathbf{V}_q\mathbf{h}^{t-1} + \mathbf{b}_q), \tag{8}$$

$$\mathbf{r}^t = \sigma(\mathbf{W}_r\boldsymbol{\xi}^t + \mathbf{V}_r\mathbf{h}^{t-1} + \mathbf{b}_r), \tag{9}$$

where $\{\mathbf{W}_q, \mathbf{V}_q, \mathbf{W}_r, \mathbf{V}_r\} \subset \mathbb{R}^{M\times M}$ are matrices whose entries are learnable weights, $\{\mathbf{b}_q, \mathbf{b}_r\} \subset \mathbb{R}^M$ are the bias parameters, and $\sigma(\cdot)$ is the sigmoid function.

The update of the hidden state $\mathbf{h}^t$ is a linear combination of the previous hidden state and the candidate state $\mathbf{c}^t$:

$$\mathbf{c}^t = \sigma(\mathbf{W}_c\boldsymbol{\xi}^t + \mathbf{V}_c(\mathbf{h}^{t-1} \odot \mathbf{r}^t) + \mathbf{b}_c), \tag{10}$$

$$\mathbf{h}^t = \mathbf{q}^t \odot \mathbf{c}^t + (1 - \mathbf{q}^t) \odot \mathbf{h}^{t-1}, \tag{11}$$

with $\odot$ being the element-wise multiplication. Similar to LSTM [81], the additive update of the hidden state can handle long-term dependencies by avoiding a quick vanishing of the errors in back-propagation, and by not overwriting important features. The GRU structure was chosen to compose the forecasting module of the proposed model because its performance is usually on par with LSTM, but with a lower computational burden [82]; nonetheless, it could be replaced by an LSTM or any other type of RNN.

III. Joint Forecasting and Interpolation of GSs

In this section, we propose an ST neural network to jointly interpolate graph nodes and forecast future signal values. More specifically, the task is to predict the future state $\mathbf{x}^{t+p}$ of a network given the history $\mathbf{X}_{\mathcal{S}}^t = \{\mathbf{x}_{\mathcal{S}}^t, ..., \mathbf{x}_{\mathcal{S}}^{t-\tau+1}\}$.[4] Thus, the input signal is a GS composed of M nodes and the output GS is a network-signal snapshot composed of $N \geq M$ nodes. We shall assume $p = 1$ to describe the proposed learning model for the time being.

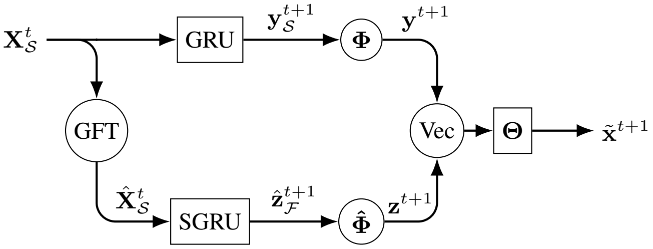

$\mathbf{X}_{\mathcal{S}}^t$
GRU
$\mathbf{y}_{\mathcal{S}}^{t+1}$
$\boldsymbol{\Phi}$
$\mathbf{y}^{t+1}$
GFT
Vec
$\boldsymbol{\Theta}$
$\tilde{\mathbf{x}}^{t+1}$
$\hat{\mathbf{X}}_{\mathcal{S}}^t$
SGRU
$\hat{\mathbf{z}}_{\mathcal{F}}^{t+1}$
$\hat{\boldsymbol{\Phi}}$
$\mathbf{z}^{t+1}$

Fig. 2: Proposed SG-GRU model. The input GS follows two routes in parallel: in the upper route, the GRU followed by interpolation is applied to the GS; in the bottom route, the GS is transformed to the frequency domain before being processed by the SGRU module and thereafter being interpolated. The outputs of these two parallel processes are stacked into a single vector, represented by operation "Vec", and fed to an FC layer.

A. Forecasting Module

The proposed learning model in Fig. 2, named spectral graph GRU (SG-GRU), combines a standard GRU cell applied to the vertex-domain GSs comprising $\mathbf{X}_{\mathcal{S}}^t$ with a GRU cell applied to the frequency-domain versions of the latter GSs comprising $\hat{\mathbf{X}}_{\mathcal{F}}^t$. The GRU acting on frequency-domain signals is named here spectral GRU (SGRU), and has the same structure as the standard GRU, except for the dimension of weight matrices and bias vectors, which are $K \times K$ and K, respectively.[5] The dimension of the hidden state is therefore K. Note that the forecasting module is applied before the interpolation. This architecture reduces the complexity of the learning model since the matrices of learnable weights have dimensions $M \times M$, for the standard GRU cell, and $K \times K$, for the SGRU, instead of $N \times N$, which would be the case if the interpolation were applied before the forecasting module.

Assuming that the entire GS $\mathbf{x}$ is $(\mathcal{F}, \epsilon)$-bandlimited, most of the information about it is expected to be stored in $\hat{\mathbf{x}}_{\mathcal{F}}$. Then, given an admissible operator $\boldsymbol{\Psi}_{\mathcal{S}}$, one has

$$\|\mathbf{x} - \boldsymbol{\Phi}_{\mathcal{S}}\boldsymbol{\Psi}_{\mathcal{S}}\mathbf{x}\|_2 \leq \frac{\epsilon}{\mathrm{SV}_{\min}(\boldsymbol{\Psi}_{\mathcal{S}}\mathbf{U}_{:,\mathcal{F}})}. \tag{12}$$

The choice of $\mathcal{F}$ will be further discussed in the experiments described in Section V.

The SGRU module in the proposed learning model predicts the (possibly time-varying) graph-frequency content of the network signals. This is key to model the underlying spatial information embedded in the graph-frequency content. Besides, it is worth pointing out that the proposed SGRU

[4] The GS at timestamp t is denoted by bold lowercase letter, $\mathbf{x}^t$, whereas the history set containing the sampled GSs in previous timestamps is denoted by bold capital letter, $\mathbf{X}_{\mathcal{S}}^t$.

[5] It is possible to use only one of the branches of the proposed structure, nonetheless preliminary experiments pointed to more promising results when using both branches combined.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3040042, IEEE Transactions on Signal and Information Processing over Networks

is different from combining the spectral graph convolution (SGC) in [83] with a GRU: in both cases, the input signal is previously transformed to the Fourier domain, $\hat{\mathbf{x}}_{\mathcal{F}}^t$, but for the SGRU, equations (8)-(10) are composed of matrix-vector multiplications, i.e., $\mathbf{W}\hat{\mathbf{x}}_{\mathcal{F}}^t$ as in a standard GRU, whereas in the latter case (SGC-GRU), equations (8)-(10) would be composed of element-wise vector multiplications, i.e., $\mathbf{w}\odot\hat{\mathbf{x}}_{\mathcal{F}}^t$, with $\mathbf{w} \in \mathbb{R}^K$. Each component of the time-varying vector $\hat{\mathbf{x}}_{\mathcal{F}}^t$, $\hat{x}_k^t$, is a time series and the SGRU is able to capture the temporal pattern, considering the correlations among different spectral components. Other graph forecasting architectures in the literature are the gated graph recurrent neural networks (GGRNN) [84] and the graph-LSTM [85]. These two learning models replace the matrix-vector multiplications of the RNN by graph convolutions using polynomial filters in the vertex domain. In [84], the graph filter is given by a polynomial on the shift operator of the graph model, whereas in [85] the graph filter is given by the Chebyshev polynomial approximation. In the SG-GRU, on the other hand, the lower branch applies a standard GRU to the frequency content of the GS.

B. GS Interpolation

The outputs from the GRU, $\mathbf{y}_{\mathcal{S}}^{t+1}$, and from the SGRU ,$\hat{\mathbf{z}}_{\mathcal{F}}^{t+1}$, are interpolated by $\mathbf{\Phi}_{\mathcal{S}}$ and $\hat{\mathbf{\Phi}}_{\mathcal{S}} = \mathbf{\Phi}_{\mathcal{S}}\mathbf{U}_{:,\mathcal{F}}$, respectively. The resulting N-dimensional vectors $\mathbf{y}^{t+1}$ and $\mathbf{z}^{t+1}$ are stacked in a single vector of size $2N$ which is processed by a fully connected (FC) layer to yield

$$\tilde{\mathbf{x}}^{t+1} = \Theta(\mathbf{y}^{t+1}, \mathbf{z}^{t+1}), \tag{13}$$

as illustrated in Fig. 2.

C. Loss Function

The loss function employed is the (empirical) mean square error (MSE). In a supervised setup, the signal values from all nodes are available for training, thus enabling the use of the entire GS $\mathbf{x}^{t+1}$ as label to compute the loss function. Given the batch size T_{b}, the loss function for supervised training is[6]

$$\mathcal{L}_{\mathrm{s}} = \frac{1}{T_{\mathrm{b}}N}\sum_{t=1}^{T_{\mathrm{b}}} \|\tilde{\mathbf{x}}^{t+1} - \mathbf{x}^{t+1}\|_2^2. \tag{14}$$

In a semi-supervised setup, on the other hand, only the sampled ground-truth signal $\mathbf{x}_{\mathcal{S}}^{t+1}$ can be accessed. In order to achieve better predictions on the unknown nodes, we propose to interpolate the sampled ground-truth signal by $\mathbf{\Phi}_{\mathcal{S}}$ before computing the MSE, yielding

$$\mathcal{L}_{\mathrm{ss}} = \frac{1}{T_{\mathrm{b}}N}\sum_{t=1}^{T_{\mathrm{b}}} \|\tilde{\mathbf{x}}^{t+1} - \mathcal{I}(\mathbf{x}_{\mathcal{S}}^{t+1})\|_2^2, \tag{15}$$

where

$$[\mathcal{I}(\mathbf{x}_{\mathcal{S}})]_n = \begin{cases} x_n, & \text{if } v_n \in \mathcal{S} \\ \left[\mathbf{\Phi}_{\mathcal{S}}\mathbf{x}_{\mathcal{S}}^{t+1}\right]_n, & \text{otherwise.} \end{cases} \tag{16}$$

The proposed model combines the interpolated temporal prediction of the sampled graph signal $\mathbf{x}_{\mathcal{S}}^t$ in both vertex and frequency domains. The spatial information provided by the underlying graph structure is used both by the interpolation and by the forecasting performed by the SGRU (restricted to the spectral support $\mathcal{F}$). The main advantages of this architecture are the reduced complexity and computational time since the loops of recurrence are applied to vectors in smaller dimensions M and K. Moreover, the model can learn during the training phase which representation, vertex or frequency domains, is more informative to forecasting. It is also worth mentioning that, different from [50], in which the interpolation is addressed only in the test phase, in the SG-GRU, the spatial prediction is taken into account in the optimization of the model's parameters in equation (16). The main limitation of the proposed learning model concerns the modeling mismatch, that is, when the graph signal is not actually approximately bandlimited with respect to the underlying graph. In this case, the SGRU may lack important information and the interpolation might be affected.

D. Computational Complexity

The SG-GRU consists of two GRU cells, called GRU and SGRU, which compute 6 matrix-vector multiplications each. The dimensions of the weight matrices in these recurrent modules applied on the vertex and frequency domains are M^2 and K^2, respectively, where K was set to $\frac{M}{3}$ in this paper (this choice will be further discussed in Section V). The input of the SGRU is the frequency-domain representation of the sampled GS restricted to the frequencies in $\mathcal{F}$, which is obtained by applying the truncated GFT matrix — the $K \times M$ matrix $\mathbf{U}_{\mathcal{S},\mathcal{F}}^{\mathrm{T}}$ — to the sampled GSs. This transform can be pre-computed, avoiding the matrix vector multiplication during the loop recurrence. In this case the input of the network becomes a signal with dimension $M + K$. The output of the GRU and the SGRU are, thereafter, interpolated by $N \times M$ and $N \times K$ matrices, respectively, which are pre-computed before running the model. Finally, an FC layer is applied to the interpolated signals, costing $2N^2$ flops. Note that the truncated GFT, the interpolations, and the FC layers are out of the recurrence loop and do not increase the computational cost if a larger sequence length τ is used. Thus, the computational cost per iteration of the SG-GRU is

$$KM + 6\tau(M^2 + K^2) + N(K + M) + 2N^2 \text{ [flops].} \tag{17}$$

IV. Applications

The proposed learning model in Fig. 2 can handle both supervised and semi-supervised scenarios. In the supervised case, measurements from the N network nodes are available in the training step but not necessarily for testing. This supervised scenario covers many different applications; a case in point is a weather station network wherein the temperature sensors are working during a period of time, but then, suddenly, some of them are shut down due to malfunctioning or maintenance cost reduction. In the semi-supervised case, on the other hand, only some nodes appear in the training set and can, therefore, be used to compute gradients. Again, the semi-supervised scenario also covers many practical applications; for instance,

[6] No regularizer or dropout layer was used.

when a sensor network is deployed with a limited number of nodes to reduce the related costs, but a finer spatial resolution is desirable, which can be obtained by a virtual denser sensor network.

Considering these two basic scenarios, we can conceive four specific types of applications:

A. *Supervised Application*

Input GS is composed of $M \leq N$ nodes but labels of all N nodes are used to compute the loss function in (14). As mentioned before, this learning model can be applied to situations in which all the N sensors are temporarily activated and, afterwards, $N - M$ sensors are turned off.

B. *Semi-supervised Application*

Both input GS and labels are composed of $M < N$. Thus, only the M in-sample are available to train the model using the loss function in (15). In this application, it is desired to predict the state of a static network with N nodes, considering that only $M < N$ sensors are deployed.

C. *Noise-corrupted Application*

Input GS is composed of all the N nodes with signals corrupted by uncorrelated additive noise, and the labels are the entire ground-truth GS. This application allows working with the proposed learning model when the sensors' measurements are not accurate. Given an $\mathcal{F}$-bandlimited GS with additive noise $\mathbf{x} = \mathbf{x}_{\rm b} + \boldsymbol{\eta}$, with $\boldsymbol{\eta}$ as in equation (6), a smoothed version of the GS can be obtained by $\tilde{\mathbf{x}}_{\rm b} = \mathbf{U}_{:,\mathcal{F}}\mathbf{U}^{\rm T}_{:,\mathcal{F}}\mathbf{x}$. Therefore, this application aims to exploit the denoising ability of the lower branch of the proposed learning model while forecasting the input GS. Note that, in this case, only the denoising capacity of the proposed model is evaluated, hence no sampling is performed over the input data.

D. *Missing-value Application*

Input GS is composed of all the N nodes but, at each time instant, a fraction of the N values measured by the sensor network are randomly chosen to be dropped (missing value). It is worth highlighting that this application is different from the (pure) supervised application in Section IV-A. In the supervised application, the set of known nodes, $\mathcal{S}$, is fixed over time, whereas the application of missing values considers different sets of known signal values at each time instant t. In other words, we have a supervised application with a time-dependent sampling set $\mathcal{S}^t$. The labels are the entire ground-truth GS. This setup evaluates the performance of the proposed SG-GRU when some of the sensors' measurements are missing, which could be due to transmission failures in a wireless network.

V. Numerical Experiments

In this section, we assess the performance of the proposed SG-GRU scheme in two real datasets. The simulation scenarios are instances of the four applications described in Section IV.

A. *Dataset Description*

The proposed learning model was evaluated on two distinct multivariate time-series datasets: temperatures provided by the Global Surface Summary of the Day Dataset (GSOD) [86], and the Seattle Inductive Loop Detector Dataset (SeattleLoop) [40].

1) Global Surface Summary of the Day Dataset: The GSOD dataset consists in daily temperature measurements in °C from 2007 to 2013, totalling $2,557$ snapshots, in 430 weather stations distributed in the United States.[7] The source provides more weather stations but only 430 worked fully from 2007 until 2013. These stations are spatially represented by a 10-nearest-neighbor graph with nonzero edge weights given by [87]:

$$A_{nm} = \frac{{\rm e}^{-(d^2_{nm}+h^2_{nm})}}{\sqrt{\sum_{j\in\mathcal{N}_n}{\rm e}^{-(d^2_{nj}+h^2_{nj})}}\sqrt{\sum_{j\in\mathcal{N}_m}{\rm e}^{-(d^2_{mj}+h^2_{mj})}}}, \quad (18)$$

in which $\mathcal{N}_n$ is the set of neighboring nodes connected to the node indexed by n, whereas d_{nm} and h_{nm} are, respectively, the geodesic distance and the altitude difference between weather stations indexed by n and m. The adjacency matrix is symmetric and the diagonal elements are set to zero.

2) Seattle Inductive Loop Detector Dataset: The SeattleLoop dataset contains traffic-state data collected from inductive loop detectors deployed on four connected freeways in the Greater Seattle area. The 323 sensor stations measure the average speed, in miles/hour, during the entire year of 2015 in a 5-minute interval, providing $105,120$ timesteps. This dataset is thus much larger than GSOD. The graph adjacency matrix provided by the source [40] is binary and the GS snapshots are barely bandlimited with respect to the graph built on this adjacency matrix. To build a network model in which the SeattleLoop time series is $(\mathcal{F}, \epsilon)$-bandlimited with a reasonably small ϵ, the nonzero entries of the binary adjacency matrix were replaced by the radial-basis function

$$A_{nm} = {\rm e}^{-\frac{\|\mathbf{x}_n-\mathbf{x}_m\|^2}{10}}, \quad (19)$$

where $\mathbf{x}_n$ and $\mathbf{x}_m$ are time series, containing 1000 time-steps, corresponding to nodes v_n and v_m, respectively.

B. *Choice of Frequency Set $\mathcal{F}$*

The larger the set $\mathcal{F}$ the more information about the input signal is considered in the model. However, the interpolation using (5) is admissible only if $|\mathcal{F}| = K \leq M$ [67]. Moreover, if K increases, the singular values of $\mathbf{U}_{\mathcal{S},\mathcal{F}}$ tend to decrease, leading to an unstable interpolation. Since the GSs considered in this paper are approximately bandlimited, using K close to M accumulates error during the training of the network. Based on the validation loss in (15), K was set to $\frac{M}{3}$.

When all nodes are available for training, that is, in the applications described in Sections IV-A, IV-C, and IV-D, $\mathcal{F}$ is chosen as the K Laplacian eigenvalues corresponding to the dominant frequency components (the ones with highest energy) of signals measured at the first 100 days. In the semi-supervised application, on the other hand, the spectral content

[7] Weather stations in the Alaska and in Hawai were not considered.

of the entire GS is unknown. Since the GSs considered in this paper are usually smooth, in the sense that most of their frequency content is supported on the indices associated with the smaller Laplacian eigenvalues, the set $\mathcal{F}$ was chosen as the K smallest eigenvalues λ_n in this scenario. The set $\mathcal{F}$ used in the application described in Section IV-B is, therefore, different from the set $\mathcal{F}$ used in the scenarios related to the applications of Sections IV-A, IV-C, and IV-D.

C. Competing Learning Techniques

Recently, many DL-based models were shown to outperform classical methods in the task of predicting ST data. Nonetheless, to the best of our knowledge, only [50] addresses the problem of predicting ST data by training a learning model with $M < N$ nodes, with the aim of reducing the training time duration. Therefore, the performance of our proposed model is here compared with DL-based models from the literature that do not actually handle sampled input GSs. Thus, we adapted the DL-based models from the literature by combining them with an interpolation strategy, such as k-NN and the GSP-based interpolator $\mathbf{\Phi}_{\mathcal{S}}$. In this context, the interpolation can be performed either: (i) before running the forecasting technique, so that the input of the competing DL-based model will be the entire GS; or (ii) after running the forecasting technique, so that the input of the competing DL-based model will be a sampled GS, thus requiring fewer learnable parameters.

We use as benchmark some LSTM-based NNs, which were shown to perform well in the strict forecasting task (i.e., time-domain prediction) on the SeattleLoop dataset in comparison with other baseline methods, such as ARIMA and SVR [31]. In addition, we also consider the ST graph convolution network (STGCN) proposed in [32] as benchmark. The graph convolutional deep learning architecture (GCDLA) [50], which allows the unknown nodes to be labeled in the test phase, serves as an additional benchmark for the semi-supervised application. In summary, the competing techniques (adapted to deal with sampled GSs) are:

(i) LSTM: simple LSTM cell;
(ii) C1D-LSTM: a 1D convolutional layer followed by an LSTM cell;
(iii) SGC-LSTM: the SGC from [83] followed by an LSTM;
(iv) TGC-LSTM: a traffic graph convolution based on the adjacency matrix combined with LSTM [31];[8]
(v) STGCN: a combination of the graph convolution from [22] with a gated-temporal convolution [32]. Hyper-parameters were set as in [32] since they lead to smaller MSE in the validation set (filter sizes were evaluated from the set $\{16, 32, 64\}$);[9]
(vi) Cheb-LSTM: an LSTM with Chebyshev polynomial filters of degree 5 in the place of matrix-vector multiplications of the LSTM equations [85, model 2];
(vii) GCDLA [50]: LSTM followed by 3 blocks of convolutional layers with polynomial of degree 2 and rough layers.[10]

As mentioned before, except for (vii), the above competing techniques do not tackle joint forecasting and interpolation tasks. Thus, they were combined with an interpolation technique. The output of methods (i)-(vi) were interpolated by $\mathbf{\Phi}_{\mathcal{S}}$, whereas a 1-hop neighborhood interpolation was applied only before the method (v), that is, each unknown value x_n^t was set as

$$[x_n^t]_{\text{unknown}} = \frac{1}{|\mathcal{N}_n|} \sum_{m \in \mathcal{N}_n} x_m^t \,. \tag{20}$$

This interpolation was combined with the STGCN in order to evaluate a non-GSP spatial prediction. The TGC-LSTM was only applied to the SeattleLoop dataset since it uses a free-flow reachability matrix, being specifically designed for traffic networks. The learning models (vi) and (vii) were evaluated only in the semi-supervised application.

D. Experimental Setup

In the applications described in Sections IV-A and IV-B, 75%, 50%, and 25% from the N nodes in $\mathcal{V}$ were selected to compose the set $\mathcal{S}$ using the greedy method in [67], with the set $\mathcal{F}_{\text{ds}}$ corresponding to the first M smallest Laplacian eigenvalues. The subscript 'ds' highlights that this is the actual frequency support of the *dataset*. This support might differ from the frequency support adopted by the learning model in Section V-B. This choice of $\mathcal{F}_{\text{ds}}$ relies on the smoothness of the underlying GS, that is, nodes near to each other are assigned with similar values. The same sampling sets were used for both supervised and semi-supervised training. All the experiments were conducted with a time window of length $\tau = 10$. The prediction length was $p = 1$ and $p = 3$ samples ahead for the GSOD dataset, that is, 1 day and 3 days, respectively, and $p = 1$ and $p = 6$ samples to SeattleLoop, that is 5 and 30 minutes, respectively.

The datasets were split into: 70% for training, 20% for validation, and 10% for test. Batch size was set to $T_{\text{b}} = 40$ and the learning rate was 10^{-4}, with step decay rate of 0.5 after every 10 epochs. Training was stopped after 100 epochs or 5 non-improving validation loss epochs. The input of the model was normalized by the maximum value in the training set. The model was trained by the RMSprop [88] with PyTorch default parameters [89]. The network was implemented in PyTorch 1.4.0 and experiments were conducted on a single NVIDIA GeForce GTX 1080.

The prediction performance was evaluated by the root mean square error (RMSE) and the mean absolute error (MAE). The error metrics MAE and RMSE have the same units as the data of interest, but RMSE is more sensitive to large errors, whereas MAE tends to treat more uniformly the prediction errors. In the semi-supervised application and in the noisy setup, the mean absolute percentage error (MAPE) was also evaluated.

E. Results: Supervised Application

TABLE III and TABLE IV show the MAE and RMSE in the supervised application. The proposed model outperformed all competitors in virtually all scenarios. When the sample size decreases, the performance gap increases compared to the

[8] Code from https://github.com/zhiyongc/Graph_Convolutional_LSTM .
[9] Code from https://github.com/VeritasYin/Project_Orion.
[10] The graph convolutions were implemented with *pytorch geometric*.

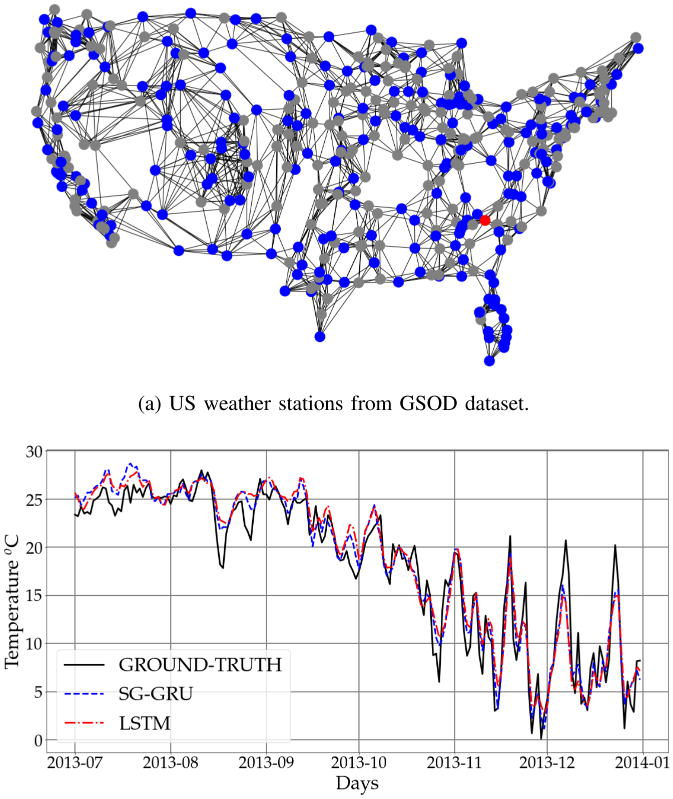
TABLE III: MAE and RMSE of supervised prediction applied to the GSOD dataset

		$M = 0.75N$		$M = 0.50N$		$M = 0.25N$	
	Methods	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
p=1	**SG-GRU**	**1.66**	**2.21**	**1.73**	**2.28**	**1.74**	**2.31**
	LSTM	2.37	3.11	2.35	3.09	2.52	3.31
	C1D-LSTM	2.32	3.02	2.40	3.15	2.66	3.49
	SGC-LSTM	3.15	4.15	3.20	4.25	3.23	4.28
	STGCN	2.20	2.98	2.44	3.29	2.40	3.22

TABLE IV: MAE and RMSE of supervised prediction applied to the SeattleLoop dataset

		$M = 0.75N$		$M = 0.50N$		$M = 0.25N$	
	Methods	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
p=1	**SGGRU**	**2.79**	**4.16**	**3.02**	**4.59**	**3.38**	**5.40**
	LSTM	3.15	4.79	3.64	5.59	4.45	7.03
	C1D-LSTM	3.25	4.95	3.70	5.70	4.49	7.08
	SGC-LSTM	3.59	5.57	3.97	6.14	4.60	7.26
	TGC-LSTM	3.03	4.59	3.54	5.45	4.40	6.98
	STGCN	**2.79**	4.32	3.11	4.82	3.65	6.10

benchmarks. On the GSOD dataset, the SG-GRU performed much better than the other strategies. We can see that, as the temperature GS is approximately ($\mathcal{F}$,ϵ)-bandlimited with small ϵ, the SG-GRU successfully captures spatial correlations by predicting the GSs' frequency content.



(a) US weather stations from GSOD dataset.

(b) Predicted temperature on a single sensor.

Fig. 3: (a) Graph of sensors in the GSOD dataset. The known (50%) and unknown (50%) nodes are colored by blue and gray, respectively. The red node, which does belong to $\mathcal{S}$, indicates the weather station whose temperature predictions are shown in (b); and (b) output of the SG-GRU and the LSTM over the ground-truth temperature in the 2$^{\text{nd}}$ semester of 2013 measured by the node highlighted in red in (a).

F. Results: Semi-supervised Application

The loss function in (15) was used for training the SG-GRU and the LSTM-based methods. For the STGCN, the interpolation of the target GS in (16) was replaced by the 1-hop interpolation. TABLE V and TABLE VI show the result of the SG-GRU and the competing approaches on the SeattleLoop and GSOD datasets, respectively. Fig. 3b shows the outputs of the SG-GRU and LSTM methods, in the second semester of 2013 over the ground-truth signal, for a weather station out of the sampling set, highlighted in Fig. 3a, considering a situation with 50% of known nodes. The SG-GRU outperformed the competing methods in the GSOD dataset. Since temperature GSs are highly smooth in the graph domain, the GSP interpolation, which is based on the assumption that the GS is bandlimited, provides good reconstruction. The GSs in the SeattleLoop dataset, on the other hand, are not as smooth as the GSs in the GSOD dataset, leading to a larger reconstruction error. Even with this limitation on the prior smoothness assumption, the SG-GRU outperformed the STGCN combined with 1-hop interpolation and the TGC-LSTM combined with GSP interpolation when the sampling set size is 25% or 50% of the total number of nodes. It is worth mentioning that the STGCN and the TGC-LSTM are learning models designed specifically for traffic forecasting. When the horizon of prediction is 30 minutes, then the SG-GRU achieved the smallest errors among all tested methods. This could be due to simultaneous ST features extraction by the SGRU module. Fig. 4 depicts the predicted speed by SG-GRU, STGCN, and TGC-LSTM for an unknown sensor with $p = 1$, $M = 0.50N$, and during the day $11/24/2015$. As can be seen, the SG-GRU was able to better fit many points in the curve. It is worth mentioning that, despite the STGCN having poorly fitted the curve in Fig. 4, it actually achieved higher accuracy on the known samples.

G. Results: Noise-corrupted Application

In many real situations, sensors' measurements can be contaminated with noise, which may worsen forecasting accuracy. Therefore, to deal with these situations, it is important to develop robust algorithms. Consider a GS $\mathbf{x}$ with standard deviation σ_x and a measurement Gaussian noise, uncorrelated across both time and graph-domain, $\boldsymbol{\eta}$ with standard deviation σ_η. The noisy GS is $\tilde{\mathbf{x}} = \mathbf{x} + \boldsymbol{\eta}$ if the whole network is measured or $\tilde{\mathbf{x}}_{\mathcal{S}} = \boldsymbol{\Psi}_{\mathcal{S}}\mathbf{x} + \boldsymbol{\eta}$, if only the subset $\mathcal{S}$ is measured.

To evaluate the robustness of the proposed learning model, both SeattleLoop and GSOD datasets were corrupted by additive Gaussian noise with zero mean and standard deviation (std) $\sigma_\eta = 0.5\sigma_x$ and $\sigma_\eta = 0.1\sigma_x$, where σ_x is the std of the entire dataset: 10°C for GSOD dataset and 12.74 miles/h for SeattleLoop dataset. In this experiment, nodes were not sampled and only the capability of handling noisy input was evaluated. TABLE VII and TABLE VIII show MAE, RMSE,

TABLE V: MAE and RMSE of semi-supervised prediction applied to the GSOD dataset

		$M = 0.75N$			$M = 0.50N$			$M = 0.25N$		
	Methods	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
p=1	**SG-GRU**	**1.77**	**2.38**	**8.91**	**1.88**	**2.53**	**9.50**	**2.06**	**2.76**	**10.7**
	LSTM	2.35	3.03	11.1	2.41	3.16	12.0	2.72	3.54	13.8
	C1D-LSTM	1.83	2.44	10.2	2.00	2.65	11.3	2.24	2.97	13.4
	SGC-LSTM	2.75	3.66	16.6	2.84	3.76	18.10	3.01	3.97	21.2
	STGCN	2.34	3.20	13.3	3.75	5.02	13.6	6.92	8.65	14.6
	STGCN-GSP	2.41	3.13	13.8	2.42	3.22	14.4	2.53	3.55	14.6
	GCDLA	3.29	4.21	15.1	3.43	4.66	17.5	5.82	7.17	23.0
	Cheb-LSTM	2.76	3.68	15.8	3.34	4.45	16.9	3.62	5.12	38.5
p=3	**SG-GRU**	**2.84**	**3.76**	**15.3**	**2.90**	**3.85**	**15.3**	**2.99**	**3.94**	**15.9**
	LSTM	2.88	3.83	16.0	2.95	3.92	17.2	3.04	4.03	17.2
	C1D-LSTM	2.88	3.84	15.8	2.96	3.92	16.3	3.05	4.03	17.1
	SGC-LSTM	3.12	4.15	17.5	3.16	4.20	18.5	3.28	4.36	20.9
	STGCN	3.33	4.40	18.97	4.28	5.53	19.7	6.95	8.48	20.7
	STGCN-GSP	3.58	4.74	20.2	3.74	4.79	18.7	4.30	5.35	19.2
	GCDLA	3.54	4.77	20.0	4.07	5.45	22.4	6.09	7.62	26.4
	Cheb-LSTM	3.17	4.24	18.2	3.41	4.62	18.5	3.44	4.67	19.0

TABLE VI: MAE and RMSE of semi-supervised approaches applied to the SeattleLoop dataset

		$M = 0.75N$			$M = 0.50N$			$M = 0.25N$		
	Methods	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
p=1	**SG-GRU**	2.98	**4.60**	7.80	**3.53**	**5.55**	**9.81**	**4.50**	**7.28**	**14.1**
	LSTM	3.06	4.73	8.50	3.61	5.66	10.5	4.56	7.34	14.6
	C1D-LSTM	3.09	4.77	8.79	3.67	5.74	10.6	4.61	7.4	14.7
	SGC-LSTM	3.46	5.38	9.92	3.86	5.99	11.5	4.65	7.44	15.1
	TGC-LSTM	3.01	4.61	9.23	3.64	**5.55**	11.3	4.82	7.75	15.1
	STGCN	**2.88**	4.65	**7.45**	3.72	6.46	11.2	5.67	10.3	18.5
	STGCN-GSP	3.01	**4.60**	**7.45**	3.47	5.50	9.64	4.53	7.28	**14.1**
	GCDLA	3.73	5.86	11.0	4.60	7.61	15.3	5.76	9.18	20.7
	Cheb-LSTM	3.31	5.09	9.10	3.72	5.77	10.7	3.87	7.58	15.6
p=6	**SG-GRU**	**3.87**	**6.18**	**11.2**	**4.18**	**6.61**	**12.6**	**4.88**	**7.77**	**15.7**
	LSTM	3.96	6.34	11.8	4.31	6.81	12.8	4.98	7.94	16.1
	C1D-LSTM	3.96	6.37	11.9	4.3	6.83	12.9	5.02	7.97	16.1
	SGC-LSTM	4.12	6.63	11.9	4.44	6.98	14.0	5.03	7.98	17.0
	TGC-LSTM	4.91	7.89	11.2	5.17	8.23	13.7	8.29	12.5	16.6
	STGCN	4.54	6.82	12.54	4.56	7.90	17.3	6.11	10.8	25.2
	STGCN-GSP	3.89	6.33	11.8	4.25	6.77	12.6	4.99	7.93	16.3
	GCDLA	4.39	6.62	10.9	5.28	8.16	15.8	6.52	9.85	17.0
	Cheb-LSTM	3.99	6.39	11.9	4.35	6.90	13.3	5.29	8.34	17.9

and MAPE[11] of the forecasting models respectively evaluated on 100 and 30 simulations of each of these noisy scenarios.

In the GSOD dataset, the proposed model achieved reasonable error levels in the presence of noisy measurements: for instance, MAE and RMSE increased 9% and 7% in comparison with the supervised application with $M = 0.75N$ when the additive noise has std $\sigma_\eta = 0.1\sigma_x$. Many GS denoising approaches are based on attenuating high frequencies of the GS [90], [91]. The SGRU module of the proposed model promotes the smoothness of the predicted GS similarly: it runs a predictive algorithm over a restricted subset of the graph frequency content, $\mathcal{F}$, and thereafter computes the inverse GFT considering only this restricted subset.

In the SeattleLoop dataset, the MAE and RMSE evaluated on the proposed model increased 4% and 2%, respectively, in comparison with the supervised application with $M = 0.75N$ when the additive noise has std $\sigma_\eta = 0.1\sigma_x$, This is a highly acceptable result, even though the STGCN achieved lower errors. The architecture of the STGCN is able to handle noisy data, especially when the horizon of prediction is small. It is also worth mentioning that the GSOD dataset is much smaller than the SeattleLoop dataset, which could also explain why the SG-GRU performs better in the GSOD dataset.

TABLE VII: MAE, RMSE and MAPE ($\%$) of forecasting applied to the GSOD with noise corruption

		$\sigma_\eta = 0.1\sigma_x$			$\sigma_\eta = 0.5\sigma_x$		
	Methods	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
p=1	**SG-GRU**	**1.81**	**2.36**	**7.70**	**2.01**	**2.61**	**8.52**
	LSTM	1.98	2.59	8.55	2.11	2.75	9.70
	C1D-LSTM	1.90	2.49	8.07	2.03	2.65	8.65
	SGC-LSTM	2.94	3.89	13.9	2.95	3.91	13.9
	STGCN	2.19	2.94	10.3	2.66	3.48	12.3
p=3	**SG-GRU**	**2.85**	**3.79**	**13.41**	**2.89**	**3.83**	**13.5**
	LSTM	2.88	3.83	13.6	2.93	3.88	13.8
	C1D-LSTM	2.86	3.8	13.4	2.91	3.85	13.6
	SGC-LSTM	3.18	4.21	15.2	3.16	4.2	15.2
	STGCN	3.17	4.23	15.6	3.21	4.25	15.7

H. Results: Missing-value Application

Another common problem in real time-series datasets are missing values, which could occur due to sensor's malfunctioning or failure in transmission. To evaluate the performance of the SG-GRU in this situation, 10% of both SeattleLoop and GSOD datasets were randomly removed (missing values).

[11] Temperatures in the GSOD dataset were converted to Fahrenheit before computing MAPE to avoid division by zero.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3040042, IEEE Transactions on Signal and Information Processing over Networks

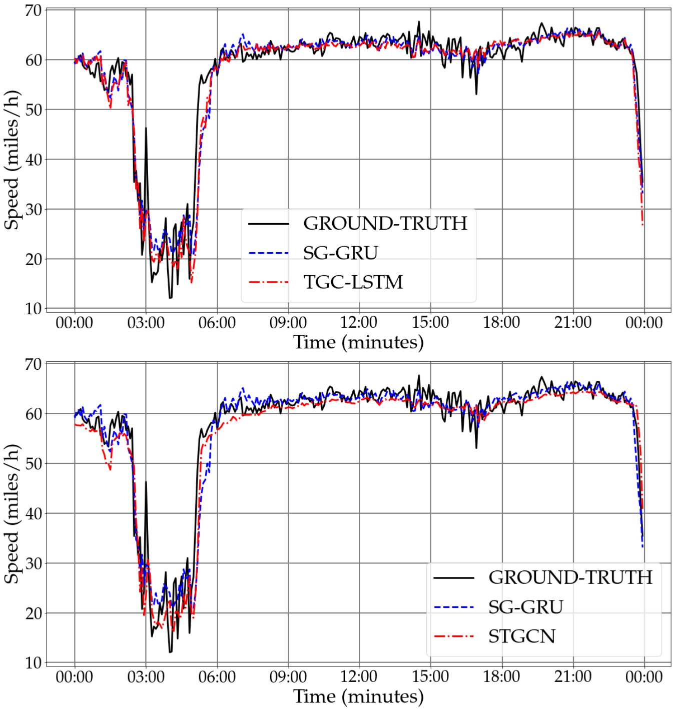


Fig. 4: Predicted signal of the sensor i005es16920 using a subset with 50% of the nodes for the SG-GRU, TGC-LSTM, and STGCN. The evaluated sensor was absent in the sampling set $\mathcal{S}$.

TABLE VIII: MAE, RMSE and MAPE (%) of forecasting applied to the SeattleLoop with noise corruption

		$\sigma_\eta = 0.1\sigma_x$			$\sigma_\eta = 0.5\sigma_x$		
	Methods	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
p=1	**SG-GRU**	2.91	4.27	13.7	3.13	4.66	16.1
	LSTM	3.21	4.85	18.2	3.45	5.20	19.4
	C1D-LSTM	3.30	5.05	19.4	3.48	5.30	20.5
	SGC-LSTM	3.96	6.28	28.3	4.07	6.44	29.9
	TGC-LSTM	2.88	4.24	13.0	3.19	4.74	14.2
	STGCN	**2.63**	**3.85**	**11.3**	**3.08**	**4.39**	**12.9**
p=6	**SG-GRU**	**3.74**	**6.02**	26.3	**3.84**	**6.19**	26.1
	LSTM	3.99	6.35	28.6	4.07	6.47	27.1
	C1D-LSTM	3.99	6.39	28.6	4.07	6.49	28.1
	SGC-LSTM	4.56	7.27	34.6	4.58	7.29	34.7
	TGC-LSTM	3.79	6.09	**26.1**	3.92	6.28	**25.3**
	STGCN	3.77	6.18	26.4	4.00	6.33	26.0

TABLE IX: MAE, RMSE and MAPE (%) of forecasting applied to the GSOD dataset with 10% of missing values

	$p = 1$			$p = 3$		
Methods	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
SG-GRU	**1.75**	**2.3**	**7.53**	**2.87**	**3.77**	**13.1**
LSTM	2.56	3.41	12.3	2.94	3.93	14.1
C1D-LSTM	2.53	3.36	11.9	2.92	3.9	13.9
SGC-LSTM	3.51	4.81	20.2	3.22	4.34	16.8
STGCN	2.10	2.87	9.97	3.22	4.31	15.2

TABLE X: MAE, RMSE and MAPE (%) of forecasting applied to the SeattleLoop dataset with 10% of missing values

	$p = 1$			$p = 6$		
Methods	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
SG-GRU	3.10	**3.85**	4.60	**6.17**	14.9	**23.7**
LSTM	3.37	4.07	5.08	6.49	19.6	27.3
C1D-LSTM	3.44	4.06	5.23	6.49	19.6	27.3
SGC-LSTM	4.01	4.58	6.34	7.31	16.3	35.1
TGC-LSTM	3.15	3.91	4.70	6.26	13.5	24.8
STGCN	**2.60**	3.91	**3.95**	6.26	**11.8**	24.9

TABLE XI: Average computational time in seconds

	SeattleLoop		GSOD	
Methods	**Training**	**Test**	**Training**	**Test**
SG-GRU	414.68	4.89	11.18	0.01
LSTM	1134.0	5.63	30.74	0.01
C1D-LSTM	1319.0	5.90	36.90	0.03
SGC-LSTM	2770.3	6.10	39.89	0.04
TGC-LSTM	1027.1	5.48	-	-
STGCN	725.58	12.6	83.92	0.12
GCDLA	1489.30	2.52	23.5	0.09
Cheb-LSTM	1099.5	20.7	34.68	0.51

Before applying the forecasting methods, the missing values were interpolated by the 1-hop interpolation in (20), thus, like in the noise-corrupted application, the input GS is not sampled. TABLE IX and TABLE X show the numerical results of this scenario considering two forecasting horizons on the GSOD and SeattleLoop datasets, respectively, each one evaluated on 100 and 30 simulations. The forecasting accuracy decreases when there are missing values, as expected. For instance, in the GSOD dataset, MAE and RMSE increased 6% and 4% in comparison with the supervised application with $M = 0.75N$. In the SeattleLoop dataset, MAE increases about 10% whereas the RMSE decreases about 8%. The GFT in the proposed model (and also in combination with the LSTM-based models) tends to smooth the output signal, reducing large deviations and consequently the RMSE. Nonetheless, it slightly increases the forecasting error across many nodes, leading to the increase in MAE.

I. Computational Cost and Efficiency

In the SeattleLoop Dataset, the epoch duration of SG-GRU was, on average, 8.5 s, whereas the more complex approaches, TGC-LSTM and STGCN, took around 40 s and 84 s per epoch, respectively. In the GSOD dataset, which is much shorter than the SeattleLoop, the average epoch duration of SG-GRU, LSTM, and STGCN were 0.20 s, 0.25 s, and 2.5 s, respectively. TABLE XI shows the average training time, including pre-processing and data preparation, as well as test phases for the 3 semi-supervised scenarios applied on the SeattleLoop and GSOD datasets, with $p = 1$. The SG-GRU required more epochs to converge than STGCN, but it still trains faster than the competing approaches.

J. Final Remarks on the Results

The consistently better results obtained by the SG-GRU for the GSOD dataset stem from the smoothness of the temperature GS with respect to the graph domain; SG-GRU relies on the assumption of bandlimited GSs. Therefore, SG-GRU is a promising approach to predict spatially smooth GSs. It is worth mentioning that the choice of the adjacency matrix is fundamental for a good performance, since it eventually defines the smoothness of the GSs. In the SeattleLoop dataset, which is not really smooth, the SG-GRU outperformed both the STGCN and the LSTM-based approaches when the sample size was small and the prediction time horizon was 30 minutes, thus indicating that the SG-GRU can capture ST dependencies by taking the network frequency content into account. Moreover, SG-GRU has low computational cost and can be boosted with more recurrent or fully connected layers, when sufficient computational resources are available.

VI. Conclusion

This work presented a new deep learning technique for jointly forecasting and interpolating network signals represented by graph signals. The proposed learning model embeds GSP tools in its basic learning-from-data unit (SG-GRU cell), thus merging model-based and deep learning approaches in a successful manner. Indeed, the proposal is able to capture spatiotemporal correlations when the input signal comprises just a small sample of the entire network. Additionally, the technique allows reliable predictions when input data is noisy or some values are missing by enforcing smoothness on the output signals. As future works, we envisage the use of the proposed SG-GRU as part of an anomaly detector in network signals, in which the anomalous sensors' measurements are characterized by large deviations from the neighboring sensors.

References

[1] A. G. Salman, B. Kanigoro, and Y. Heryadi, "Weather forecasting using deep learning techniques," in *International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, Oct 2015, pp. 281–285.

[2] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Transactions on Intelligence Transportation Systems*, vol. 16, no. 2, pp. 865–873, Apr 2015.

[3] S. M. Smith, K. L. Miller, G. Salimi-Khorshidi, M. Webster, C. F. Beckmann, T. E. Nichols, J. D. Ramsey, and M. W. Woolrich, "Network modelling methods for FMRI," *NeuroImage*, vol. 54, no. 2, pp. 875–891, Jan 2011.

[4] L. Li, K. Ota, and M. Dong, "When weather matters: IoT-based eletric. load forecasting for smart grid," *IEEE Communications Magazine*, vol. 55, no. 10, pp. 46–51, Oct 2017.

[5] E. Racah, C. Beckham, T. Maharaj, S. Ebrahimi Kahou, M. Prabhat, and C. Pal, "ExtremeWeather: A large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events," in *Advances in Neural Information Processing Systems 30 (NIPS)*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., Dec 2017, pp. 3402–3413.

[6] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 664–672, Oct 2003.

[7] L. Cai, Z. Zhang, J. Yang, Y. Yu, T. Zhou, and J. Qin, "A noise-immune Kalman filter for short-term traffic flow forecasting," *Physica A: Statistical Mechanics and its Applications*, vol. 536, p. 122601, Dec 2019.

[8] S. R. Chandra and H. Al-Deek, "Predictions of freeway traffic speeds and volumes using vector autoregressive models," *Journal of Intelligence Transportation Systems*, vol. 13, no. 2, pp. 53–72, Apr 2009.

[9] O. Kramer and F. Gieseke, "Short-term wind energy forecasting using support vector regression," in *Soft Computer Models in Industrial and Environmental Applications,* 6th *International Conference (SOCO)*, vol. 87. Springer, 2011, pp. 271–280.

[10] G. Leshem and Y. Ritov, "Traffic flow prediction using AdaBoost algorithm with random forests as a weak learner," *Journal International Journal of Intelligence Technology*, vol. 2, pp. 1305–6417, Jan 2007.

[11] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: deep belief networks with multitask learning," *IEEE Transactions on Intelligence Transportation Systems*, vol. 15, no. 5, pp. 2191–2201, Apr 2014.

[12] Yuhan Jia, Jianping Wu, and Yiman Du, "Traffic speed prediction using deep learning method," in *IEEE International Conference on Intelligence Transportation Systems (ITSC)*, Dec 2016, pp. 1217–1222.

[13] A. Salman, Y. Heryadi, E. Abdurahman, and W. Suparta, "Weather forecasting using merged long short-term memory model (LSTM) and autoregressive integrated moving average (arima) model," *Journal of Comput. Science*, vol. 14, pp. 930–938, July 2018.

[14] S. Liang, L. Nguyen, and F. Jin, "A multi-variable stacked long-short term memory network for wind speed forecasting," in *IEEE International Conference on Big Data*, Dec 2018, pp. 4561–4564.

[15] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MA, USA,USA: MIT press, 2016.

[16] X. SHI, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. WOO, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 802–810.

[17] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks," *Sensors*, vol. 17, no. 7, pp. 1–16, June 2017.

[18] C. J. Huang and P. H. Kuo, "A deep CNN-LSTM model for particulate matter $(\mathrm{Pm}_{2.5})$ forecasting in smart cities," *Sensors*, vol. 18, no. 2220, July 2018.

[19] Y. Wu, H. Tan, L. Qin, B. Ran, and Z. Jiang, "A hybrid deep learning based traffic flow prediction method and its understanding," *Transportation Research Part C: Emerging Technologies*, vol. 90, pp. 166–180, May 2018.

[20] W. Li, W. Tao, J. Qiu, X. Liu, X. Zhou, and Z. Pan, "Densely connected convolutional networks with attention LSTM for crowd flows prediction," *IEEE Access*, vol. 7, pp. 140 488–140 498, Sep 2019.

[21] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," *eprint arXiv:1506.05163*, June 2015.

[22] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 3844–3852.

[23] R. Levie, F. Monti, X. Bresson, and M. M. Bronstein, "CayleyNets: Graph convolutional neural networks with complex rational spectral filters," *IEEE Transactions on Signal Processing*, vol. 67, no. 1, pp. 97–109, Nov 2019.

[24] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *eprint arXiv:1609.02907*, Sep 2016.

[25] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in 6th *International Conference on Learning Representations (ICLR)*, Fev 2018, pp. 1–16.

[26] L. Zhao, Y. Song, M. Deng, and H. Li, "Temporal graph convolutional network for urban traffic flow prediction method," *eprint arXiv:1811.05320*, Dec 2018.

[27] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung, "GaAN: gated attention networks for learning on large and spatiotemporal graphs," *eprint arXiv:1803.07294*, Mar 2018.

[28] B. Wang, X. Luo, F. Zhang, B. Yuan, A. L. Bertozzi, and P. J. Brantingham, "Graph-based deep modeling and real time forecasting of sparse spatio-temporal data," *eprint arXiv:1804.00684*, Apr 2018.

[29] X. Geng, Y. Li, L. Wang, L. Zhang, Q. Yang, J. Ye, and Y. Liu, "Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 3656–3663, July 2019.

[30] W. Chen, L. Chen, Y. Xie, W. Cao, Y. Gao, and X. Feng, "Multi-range attentive bicomponent graph convolutional network for traffic forecasting," *eprint arXiv:1911.12093*, Nov 2019.

[31] Z. Cui, K. Henrickson, R. Ke, and Y. Wang, "Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting," *IEEE Transactions on Intelligence Transportation Systems*, Nov 2019.

[32] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, July 2018, pp. 3634–3640.

[33] M. Wang, B. Lai, Z. Jin, Y. Lin, X. Gong, J. Huang, and X. Hua, "Dynamic spatio-temporal graph-based cnns for traffic prediction," *eprint arXiv:1812.02019*, Mar 2020.

[34] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph WaveNet for deep spatial-temporal graph modeling," *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1907–1913, Aug 2019.

[35] K. Lee and W. Rhee, "DDP-GCN: Multi-graph convolutional network for spatiotemporal traffic forecasting," *eprint arXiv:1905.12256*, May 2019.

[36] Z. Diao, X. Wang, D. Zhang, Y. Liu, K. Xie, and S. He, "Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, July 2019, pp. 890–897.

[37] S. Fang, Q. Zhang, G. Meng, S. Xiang, and C. Pan, "GstNet: Global spatial-temporal network for traffic flow prediction," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, Aug 2019, pp. 10–16.

[38] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," https://github.com/wanhuaiyu/STSGCN/blob/master/paper/AAAI2020-STSGCN.pdf, 2020.

[39] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, July 2019, pp. 922–929.

[40] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction," *eprint arXiv:1801.02143*, Jan 2018.

[41] X. Wang, C. Chen, Y. Min, J. He, B. Yang, and Y. Zhang, "Efficient metropolitan traffic prediction based on graph recurrent neural network," *arXiv:1811.00740*, Nov 2018.

[42] B. Liao, D. McIlwraith, J. Zhang, T. Chen, C. Wu, S. Yang, Y. Guo, and F. Wu, "Deep sequence learning with auxiliary information for traffic prediction," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 537–546, July 2018.

[43] B. Liao, J. Zhang, M. Cai, S. Tang, Y. Gao, C. Wu, S. Yang, W. Zhu, Y. Guo, and F. Wu, "Dest-ResNet: A deep spatiotemporal residual network for hotspot traffic speed prediction," in *Proceedings of the* 26th *ACM International Conference on Multimedia*, Oct 2018, pp. 1883–1891.

[44] D. Han, J. Chen, and J. Sun, "A parallel spatiotemporal deep learning network for highway traffic flow forecasting," *International Journal of Distributed Sensor Networks*, vol. 15, no. 2, Feb 2019.

[45] Z. Lv, J. Xu, K. Zheng, H. Yin, P. Zhao, and X. Zhou, "LC-RNN: A deep learning model for traffic speed prediction," in *International Joint Conference on Artificial Intelligence (IJCAI)*, July 2018, pp. 3470–3476.

[46] Y. Lin, X. Dai, L. Li, and F. Y. Wang, "Pattern sensitive prediction of traffic flow based on generative adversarial framework," *IEEE Transactions on Intelligence Transportation Systems*, vol. 20, no. 6, pp. 2395–2400, Aug 2019.

[47] H. F. Yang, T. S. DIllon, and Y. P. P. Chen, "Optimized structure of the traffic flow forecasting model with a deep learning approach," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2371–2381, Oct 2017.

[48] K. Zhang, L. Zheng, Z. Liu, and N. Jia, "A deep learning based multitask model for network-wide traffic speed prediction," *Neurocomputing*, vol. 396, pp. 438 – 450, Apr 2020.

[49] X. Ouyang, C. Zhang, P. Zhou, H. Jiang, and S. Gong, "Deepspace: An online deep learning framework for mobile big data to understand human mobility patterns," *arXiv:1610.07009*, Mar 2016.

[50] M. Khodayar and J. Wang, "Spatio-temporal graph deep neural network for short-term wind speed forecasting," *IEEE Transactions on Sustainable Energy*, vol. 10, no. 2, pp. 670–681, Apr 2019.

[51] M. Khodayar, O. Kaynak, and M. E. Khodayar, "Rough deep neural architecture for short-term wind speed forecasting," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 6, pp. 2770–2779, July 2017.

[52] Q. Zhu, J. Chen, L. Zhu, X. Duan, and Y. Liu, "Wind speed prediction with spatio-temporal correlation: A deep learning approach," *Energies*, vol. 11, p. 705, Mar 2018.

[53] C. Y. Zhang, C. L. Chen, M. Gan, and L. Chen, "Predictive deep boltzmann machine for multiperiod wind speed forecasting," *IEEE Transactions on Sustainable Energy*, vol. 6, no. 4, pp. 1416–1425, July 2015.

[54] M. Khodayar, S. Mohammadi, M. E. Khodayar, J. Wang, and G. Liu, "Convolutional graph autoencoder: A generative deep neural network for probabilistic spatio-temporal solar irradiance forecasting," *IEEE Transactions on Sustainable Energy*, vol. 11, no. 2, pp. 571–583, Apr 2020.

[55] S. Rasp and S. Lerch, "Neural networks for postprocessing ensemble weather forecasts," *Monthly Weather Review*, vol. 146, no. 11, pp. 3885–3900, Oct 2018.

[56] C. Li, Z. Cui, W. Zheng, C. Xu, and J. Yang, "Spatio-temporal graph convolution for skeleton based action recognition," in *Proceedings of the AAAI Conference on Artificial Intelligence*, Apr 2018.

[57] C. Wu, X.-J. Wu, and J. Kittler, "Spatial residual layer and dense connection block enhanced spatial temporal graph convolutional network for skeleton-based action recognition," in *The IEEE International Conference on Comput. Vision (ICCV) Workshops*, Oct 2019.

[58] C. Si, W. Chen, W. Wang, L. Wang, and T. Tan, "An attention enhanced graph convolutional LSTM network for skeleton-based action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[59] S. Gadgil, Q. Zhao, E. Adeli, A. Pfefferbaum, E. V. Sullivan, and K. M. Pohl, "Spatio-temporal graph convolution for functional MRI analysis," *eprint arXiv:2003.10613*, Mar 2020.

[60] H. Huang, X. Hu, J. Han, J. Lv, N. Liu, L. Guo, and T. Liu, "Latent source mining in FMRI data via deep neural network," *IEEE International Symposium on Biomedical Imaging (ISBI)*, pp. 638–641, July 2016.

[61] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," *arXiv:1511.05493*, Sep 2015.

[62] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, May 2018.

[63] S. K. Narang, A. Gadde, E. Sanou, and A. Ortega, "Localized iterative methods for interpolation in graph structured data," in *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Dec 2013, pp. 491–494.

[64] J. Chen, H.-r. Fang, and Y. Saad, "Fast approximate knn graph construction for high dimensional data via recursive lanczos bisection." *Journal of Machine Learning Research*, vol. 10, no. 9, Sep 2009.

[65] S. Segarra, A. G. Marques, G. Leus, and A. Ribeiro, "Interpolation of graph signals using shift-invariant graph filters," in *23rd European Signal Processing Conference (EUSIPCO)*, Aug 2015, pp. 210–214.

[66] I. Pesenson, "Sampling in Paley-Wiener spaces on combinatorial graphs," *Transactions of the American Mathematical Society*, vol. 360, no. 10, pp. 5603–5627, May 2008.

[67] S. Chen, R. Varma, A. Sandryhaila, and J. Kovačević, "Discrete signal processing on graphs: Sampling theory," *IEEE Transactions on Signal Processing*, vol. 63, no. 24, pp. 6510–6523, Aug 2015.

[68] S. K. Narang and A. Ortega, "Local two-channel critically sampled filter-banks on graphs," *IEEE International Conference on Image Processing*, pp. 333–336, Sep 2010.

[69] M. J. Spelta and W. A. Martins, "Online temperature estimation using graph signals," *XXXVI Simpósio Brasileiro de Telecomunicaçoes e Processamento de Sinais (SBrT)*, pp. 154–158, Sep 2018.

[70] M. J. M. Spelta and W. A. Martins, "Normalized LMS algorithm and data-selective strategies for adaptive graph signal estimation," *Signal Processing*, vol. 167, p. 107326, Feb 2020.

[71] K. Manohar, B. W. Brunton, J. N. Kutz, and S. L. Brunton, "Data-driven sparse sensor placement for reconstruction: Demonstrating the benefits of exploiting known patterns," *IEEE Control Systems Magazine*, vol. 38, no. 3, pp. 63–86, June 2018.

[72] D. Romero, V. N. Ioannidis, and G. B. Giannakis, "Kernel-based reconstruction of space-time functions on dynamic graphs," *IEEE Journal on Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 856–869, Sep 2017.

[73] E. Isufi, A. Loukas, N. Perraudin, and G. Leus, "Forecasting time series with varma recursions on graphs," *IEEE Transactions on Signal Processing*, vol. 67, no. 18, pp. 4870–4885, 2019.

[74] P. Di Lorenzo, S. Barbarossa, P. Banelli, and S. Sardellitti, "Adaptive least mean squares estimation of graph signals," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 4, pp. 555–568, Sep 2016.

[75] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 61, no. 7, pp. 1644–1656, Jan 2013.

[76] S. K. Narang and A. Ortega, "Downsampling graphs using spectral theory," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar 2011, pp. 4208–4211.

[77] P. Lorenzo, S. Barbarossa, and P. Banelli, *Chapter 9 - Sampling and Recovery of Graph Signals*, P. M. Djurić and C. Richard, Eds. Academic Press, 2018.

[78] S. Chen, R. Varma, A. Singh, and J. Kovačević, "Signal recovery on graphs: Fundamental limits of sampling strategies," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 4, pp. 539–554, Oct 2016.

[79] B. J. Winer, "Statistical principles in experimental design," *McGraw-Hill Book Company*, 1962.

[80] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *eprint arXiv:1409.1259*, Sep 2014.

[81] A. Graves, "Generating sequences with recurrent neural networks," *eprint arXiv:1308.0850*, Aug 2013.

[82] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv:1412.3555*, Dec 2014.

[83] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *eprint arXiv:1312.6203*, Dec 2013.

[84] L. Ruiz, F. Gama, and A. Ribeiro, "Gated graph recurrent neural networks," *preprint arXiv:2002.01038*, Feb 2020.

[85] Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson, "Structured sequence modeling with graph convolutional recurrent networks," in *International Conference on Neural Information Processing*. Springer, Nov. 2018, pp. 362–373.

[86] National climactic data center. (202001, Feb. 1). [Online]. Available: ftp://ftp.ncdc.noaa.gov/pub/data/gsod

[87] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Frequency analysis," *IEEE Transactions on Signal Processing*, vol. 62, no. 12, pp. 3042–3054, Apr 2014.

[88] G. Hinton, N. Srivastava, and K. Swersky, "Lecture 6e rmsprop: Divide the gradient by a running average of its recent magnitude," 2012.

[89] Pytorch RMSprop. (2020, Feb. 1). [Online]. Available: https://pytorch.org/docs/stable/optim.html?highlight=rms#torch.optim.RMSprop

[90] D. I. Shuman, P. Vandergheynst, and P. Frossard, "Chebyshev polynomial approximation for distributed signal processing," in *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, June 2011, pp. 1–8.

[91] N. Tremblay and P. Borgnat, "Subgraph-based filterbanks for graph signals," *IEEE Transactions on Signal Processing*, vol. 64, no. 15, pp. 3827–3840, Mar 2016.

Gabriela Lewenfus received the Applied Mathematics bachelor's degree from the Federal University of Rio de Janeiro (UFRJ, Brazil) in 2018, and the M.Sc. degree in Electrical Engineering also from UFRJ in 2020. She is currently working as data scientist. Her research interests are in the fields of machine learning, graph signal processing applications, and time series analysis with focus on anomaly detection.

Wallace A. Martins (S'07-M'12-SM'20) received the Electronics Engineer degree from the Federal University of Rio de Janeiro (UFRJ, Brazil) in 2007, and both the M.Sc. and D.Sc. degrees in Electrical Engineering also from UFRJ in 2009 and 2011, respectively. He was a Research Visitor at University of Notre Dame (USA, 2008), at Université de Lille 1 (France, 2016), and at Universidad de Alcalá (Spain, 2018). Since 2013 he has been with the Department of Electronics and Computer Engineering (DEL/Poli) and Electrical Engineering Program (PEE/COPPE) at UFRJ, where he is presently a tenured Associate Professor (on leave). He served as Academic Coordinator and Deputy Department Chairman (DEL/Poli) in the period 2016-2017 at UFRJ. Before joining UFRJ, he worked as Associate Professor at CEFET/RJ (Brazil, 2010-2013). He is currently a Research Associate working with the Interdisciplinary Centre for Security, Reliability and Trust (SnT) at Université du Luxembourg. He is a member (Associate Editor) of the editorial board for the IEEE Signal Processing Letters. His research interests are in the fields of digital signal processing, especially adaptive signal processing and graph signal processing, as well as telecommunications, with focus on equalization and precoding for wireless communications. Dr. Martins received the Best Student Paper Award from EURASIP at EUSIPCO-2009, Glasgow, Scotland, and the 2011 Best Brazilian D.Sc. Dissertation Award from Capes.

Symeon Chatzinotas (S'06–M'09–SM'13) received the M.Eng. degree in telecommunications from the Aristotle University of Thessaloniki, Thessaloniki, Greece, in 2003, and the M.Sc.and Ph.D. degrees in electronic engineering from the University of Surrey, Surrey, U.K., in 2006 and 2009, respectively. He is currently a Full Professor/Chief Scientist I and Co-Head of the SIGCOM Research Group with SnT, University of Luxembourg.

Prof. Chatzinotas was a Visiting Professor with the University of Parma, Italy, and he was involved in numerous research and development projects for the National Center for Scientific Research Demokritos, the Center of Research and Technology Hellas, and the Center of Communication Systems Research, University of Surrey.

He has co-authored more than 400 technical papers in refereed international journals, conferences, and scientific books. He was a co-recipient of the 2014 IEEE Distinguished Contributions to Satellite Communications Award, the CROWNCOM 2015 Best Paper Award, and the 2018 EURASIC JWCN Best Paper Award. He is currently the editorial board of the IEEE Open Journal of Vehicular Technology and the International Journal of Satellite Communications and Networking.

Björn Ottersten (S'87–M'89–SM'99–F'04) was born in Stockholm, Sweden, in 1961. He received the M.S. degree in electrical engineering and applied physics from Linköping University, Linköping, Sweden, in 1986, and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, USA, in 1990. He is currently the Director of the Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg.

Prof. Ottersten has held research positions with the Department of Electrical Engineering, Linköping University, the Information Systems Laboratory, Stanford University, the Katholieke Universiteit Leuven, Leuven, Belgium, and the University of Luxembourg, Luxembourg. From 1996 to 1997, he was the Director of Research with ArrayComm, Inc., a start-up in San Jose, CA, USA, based on his patented technology. In 1991, he was appointed a Professor of signal processing with the Royal Institute of Technology, Stockholm, Sweden. From 1992 to 2004, he was the Head of the Department for Signals, Sensors, and Systems, KTH, and from 2004 to 2008, was the Dean of the School of Electrical Engineering.

He was a recipient of the IEEE Signal Processing Society Technical Achievement Award in 2011 and twice a recipient of the European Research Council Advanced Research Grant in 2009–2013 and in 2017–2022. He has co authored journal papers that received the IEEE Signal Processing Society Best Paper Award in 1993, 2001, 2006, and 2013, and seven IEEE Conference Papers Best Paper Awards. He has served as an Associate Editor for the IEEE Transactions on Signal Processing and the Editorial Board for the IEEE Signal Processing Magazine. He is currently a member of the Editorial Boards of