Complex-Value Spatio-temporal Graph Convolutional Neural Networks and its Applications to Electric Power Systems AI

Tong Wu, *Member, IEEE*, Anna Scaglione, *Fellow, IEEE*, Daniel Arnold, *Member, IEEE*,

arXiv:2208.08485v2 [cs.LG] 22 Sep 2022

***Abstract*—The effective representation, precessing, analysis, and visualization of large-scale structured data over graphs are gaining a lot of attention. So far most of the literature considered exclusively real-valued signals. However, signals are often sparse in the Fourier domain, and more informative and compact representations for them can be obtained using the complex envelope of their spectral components, as opposed to the original real-valued signals. Motivated by this fact, in this work we generalize graph convolutional neural networks (GCN) to the complex domain, deriving the theory that allows to incorporate a complex-valued graph shift operators (GSO) in the definition of graph filters (GF) and process complex-valued graph signals (GS). The theory developed is generalized to handle spatio-temporal complex network processes. We prove that complex-valued GCNs can be stable with respect to perturbations of the underlying graph support, by bounding of the error propagation through multiple NN layers. Then we apply complex GCN to power grid state forecasting, power grid cyber-attack detection and localization and demonstrate their superior performance relative to several benchmarks.**

***Index Terms*—Graph Neural Networks, Power System State Forecasting, False Data Localization.**

I. Introduction

In machine learning (ML) applications in which signals have a sparse spectrum, the best representation for signals is through their complex envelopes. This explains the popularity of complex-valued neural networks (Cplx-NN), introduced in the seminal paper [1], in a number of different domains, such as physical layer communications, biological signals, array processing etc. (see [2] for a review of the theory behind Cplx-NN and [3] for a survey of its main applications). Motivated primarily by the application of Artificial Intelligence (AI) in electric power systems, the overarching goal of this paper is to extend the benefits of Cplx-NN to the analysis of complex graph signals, first introducing Complex Graph Convolutional Neural Networks (Cplx-GCNs), and then investigating their potential benefits in processing electric power systems measurements. In fact, in power systems the AC voltage at each node (called bus) is concentrated around 60 or 50 Hz and the vector of complex envelopes of the voltage signals (called phasors) represents the state of the electric power network. The abundance of high-quality estimates of the voltage and current phasors acquired using phasor measurement units (PMUs) has already spurred interest in complex graph signal processing (GSP) as a framework to process them and interpret their properties [4]. Complex GSP is the most natural framework to analyze the power system state because it has a physical interpretation rooted in Ohm's law [4].

GSP is a vibrant branch of signal processing research whose aim is to generalize digital signal process (DSP) notions, and Fourier analysis in particular, for data supported on graphs [5]. A graph signal (GS) is a vector indexed by the node set of a weighted graph, representing both the data (node attributes) and the underlying structure (edge attributes). The cornerstone of GSP is the definition of Graph Shift Operator (GSO)[1]. The vast majority of GSP-based algorithms uses real-valued GSOs and considers real-valued graph signals (GS) (see e.g. the surveys [6, 7]). Having selected the GSO, one can define graph-filters; the most popular graph filter model is the Chebyshev filter [5, 8, 9]. The development of complex valued GSP has received far less attention. In addition to power systems [4, 10], complex GSP algorithms have been found applications, for example, in wireless communication networks [11] and sensor networks [12]. In power systems, complex GSP is the most natural framework [4] since using the complex system matrix as the Graph Shift Operator (GSO) has a physical interpretation rooted in Ohm's law. The caveat is that the GSO, which is the admittance matrix, is only symmetric, not conjugate symmetric, which, as we later discuss in Section III-A, requires some special care.

GSP algorithms that rely only on linear models have limited representation capability. Interestingly, the first instance of Graph Neural Networks (GNN) architecture appeared well ahead of the development of GSP [13]. The early models of GNN can be interpreted as a special case of the more general design introduced in [14], where the authors extend the Convolutional NN (CNN) model using graph filters. To process time-series of graph signals, whose samples are not independent and identically distributed (i.i.d.), the most effective architectures are Spatio-Temporal versions of this idea, such as (STGCN) (see e.g. [15, 16] which are early works on the subject) and Graph Recursive NN (GRN) (first proposed in [17]). In a nutshell, their design includes feed-forward and feedback graph-temporal filters in each layer. A thorough analysis of the stability of these designs is in [18], which inspired the stability analysis in this paper. Real-valued GCNs have showcased strong generalization capability in high-dimensional state spaces, learning complicated tasks with lower prior knowledge [19].

To the best of our knowledge, thus far, GCNs (and its variants) have been studied and applied only in the real domain (see e.g. [19] for a review). The construction of complex GCN we study in this paper follows exactly the same logic of cascading layers of complex graph-temporal filters with nonlinear activation functions for complex data.

Tong Wu, and Anna Scaglione are with the department of Electrical and Computer Engineering at Cornell Tech Campus, Cornell University, NY, USA (e-mail: {tw385, as337}@cornell.edu). Daniel Arnold is with Lawrence Berkeley National Laboratory (e-mail: dbarnold@lbl.gov).

[1]The name comes from the fact that originally the GSO was a generalization of the z variable, corresponding to a time shift in the z transform, although the definition often selects the Laplacian of the network graph, which is a graph signal differential operator.

Spatio-temporal graph convolutional neural networks (Cplx-STGCN) are applicable not only to power systems, but to any networked system where nodal signals and their interactions can be modeled effectively as a vector of envelopes for its spectral components. Prior to summarizing our contributions, next we provide a brief review of the literature on real-valued GCN for power systems applications, including the ones that we consider in our experiments to test the Cplx-STGCN performance.

A. Related Works

Several papers have already applied real-valued GCN to power systems' data analysis and management [20]. Applications include, for example, fault localization [21], power system state estimation [22], anomaly detection [23, 24], detection and localization of stealth false data injection (FDI) attacks, synthetic feeder generation [25], to name a few.

The two applications we choose to test numerically Cplx-GCN architectures are that of detection and localization of FDI attacks and power systems state estimation and forecasting (PSSE and PSSF). We note that PSSF has so far been pursued via single-hidden-layers NNs [26, 27], and further investigated by the Recurrent neural networks in [28] and Graph Recurrent neural networks [29]. The state-of-art neural network algorithms for FDI attack detection have been pursued by the Chebyshev GCN[30], CNN [31] and RNN [32].

All works on real-GCN for power systems have in common the following limitations: 1) they ignore the correlation among real and imaginary parts of power systems signals and use real GSO; 2) they do not consider temporal correlation of voltage phasors samples.

B. Contributions

The aim of this paper is to establish the framework of complex-valued STGCN and elucidate how they can be applied to power grid signals inference problems. Our main contributions are as follows:

- We combine the ideas in [1] and [14] and generalize the training of graph convolutional neural networks (GCN) so that they can operate complex domain, with complex-valued graph shift operators (GSO) and complex-valued graph signals.
- We provide analytical bounds for the impact of perturbations in the GSO, and derive bounds for how the error propagates through the multi-layer GNN structure.
- We further extend GCN to process streaming data through Cplx-STGCN architectures.
- We show how to apply correctly this framework to power systems. This entails choosing as input the voltage phasors signals, the admittance matrix as our GSO, and the Graph Fourier basis suggested in [4].
- We show that our method outperforms the prior art in detecting and localizing FDI attacks as well as in PSSE and PSSF mean squared error (MSE) performance. We also empirically evaluate the sensitivity of the architecture to model changes.

The rest of the paper is organized as follows. In Section II, we briefly review the key notions of GSP, setting the stage in Section III where we derive the physics inspired GSO and introduce our graph neural networks architectures whose sensitivity is analyzed in Section IV. In Section V, we describe two applications of the proposed GCN and GRN frameworks that are tested numerically in Section VI. Finally, we conclude the paper in Section VII.

II. Preliminaries on Power Systems

The electric grid network has an associated undirected weighted graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ where nodes are *buses* and its edges are its *transmission lines*. The edge weights $\forall(i,j) \in \mathcal{E}$ are branch admittances $y_{ij} \in \mathbb{C}$ and each node has a shunt admittance $y_{ii}^{sh}, i \in \mathcal{V}$. With these parameters one can define the *system admittance matrix* $\boldsymbol{Y} \in \mathbb{C}^{|\mathcal{V}|\times|\mathcal{V}|}$ that is defined as

$$[\boldsymbol{Y}]_{i,j} = \begin{cases} y_{ii}^{sh} + \sum_{k\in\mathcal{N}_i} y_{i,k}, i = j \\ -y_{i,j}, i \neq j \end{cases} \tag{1}$$

Kichhoff's and Ohm's laws relate the current and voltage phasors for the entire network as follows:

$$\boldsymbol{i} = \boldsymbol{Y}\boldsymbol{v},\ v_n = |v_n|\ e^{\mathrm{j}\varphi_n^v},\ i_n = |i_n|\ e^{\mathrm{j}\varphi_n^c},\ \forall n \in \mathcal{V}, \tag{2}$$

where $\boldsymbol{v}$ and $|\boldsymbol{v}|$ are the vectors of bus voltage phasors and magnitudes, respectively, with $\boldsymbol{v} \in \mathbb{C}^{|\mathcal{V}|\times 1}$ and $|\boldsymbol{v}| \in \mathbb{R}_+^{|\mathcal{V}|\times 1}$, $\mathrm{j} = \sqrt{-1}$ is the imaginary unit and $\boldsymbol{i} \in \mathbb{C}^{|\mathcal{V}|\times 1}$ and $|\boldsymbol{i}| \in \mathbb{R}_+^{|\mathcal{V}|\times 1}$ denote the vectors of net bus current phasors and magnitudes. Ohm's law allows us to view voltage as the output *low-pass* filter by $\boldsymbol{v} = \boldsymbol{Y}^{-1}\boldsymbol{i}$. Let $\boldsymbol{s} = \boldsymbol{p} + \mathrm{j}\boldsymbol{q}$ be the vector of net apparent power at buses ($\boldsymbol{s} = [s_1, \cdots, s_{|\mathcal{V}|}]^\top$), with the n^{th} entry $s_n = p_n + \mathrm{j}q_n$, where p_n and q_n are the active and reactive power injection at bus n, respectively.

The vector of net apparent power injections is:

$$\boldsymbol{s} = \boldsymbol{v} \odot \boldsymbol{i}^* = \boldsymbol{v} \odot (\boldsymbol{Y}\boldsymbol{v})^*, \tag{3}$$

where $\odot$ is the Hadamard product and $\boldsymbol{i}^*$ is the conjugate of a complex vector $\boldsymbol{i}$. $\boldsymbol{s} = \boldsymbol{p} + \mathrm{j}\boldsymbol{q}$ and the n^{th} entries of the real and imaginary parts p_n and q_n are the active and reactive power injection at bus n, respectively.

The appropriate grid graph shift operator (GSO) $\mathbf{S}$ is the system admittance matrix, $\mathbf{S} = \boldsymbol{Y}$. Note that unlike the graph Laplacian, this GSO $\mathbf{S} = \boldsymbol{Y}$ is invertible (albeit ill-conditioned), thanks to the diagonal component of the shunt admittances[2].

III. Complex-Valued Spatio-Temporal Graph Convolution Neural Network

A. Grid-Graph Signal Processing

In our work the graph signal $\boldsymbol{x} \in \mathbb{R}^{|\mathcal{V}|}$ is a vector of voltage phasors at each bus, an $[\boldsymbol{x}]_i, \forall i \in \mathcal{V}$ is the i-th entry of this state vector. The set $\mathcal{N}_i$ denotes the subset of nodes connected to node i. A graph shift operator (GSO) is a matrix $\mathbf{S} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ that linearly combines graph signal neighbors' values. Almost all operations including filtering,

[2]The magnitudes of the shut admittaces y_{ii}^{sh} are very small relative to the line admittances y_{ij} and are often neglected

transformation and prediction are directly related to the GSO. In this work, we focus on complex symmetric GSOs, i.e. such that $\mathbf{S} = \mathbf{S}^\top$ that are appropriate for our power grid application where $\mathbf{S} = \boldsymbol{Y}$. A graph filter is a linear matrix operator $\mathcal{H}(\mathbf{S})$ that is a function of the GSO and operates on graph signals as follows

$$\boldsymbol{w} = \mathcal{H}(\mathbf{S})\boldsymbol{x}. \tag{4}$$

What defines the dependency of $\mathcal{H}(\mathbf{S})$ on the GSO is that $\mathcal{H}(\mathbf{S})$ must be shift-invariant (like a linear time invariant filter in the time domain), i.e. matrix operators such that $\mathbf{S}\mathcal{H}(\mathbf{S}) \equiv \mathcal{H}(\mathbf{S})\mathbf{S}$. This property is satisfied if and only if $\mathcal{H}(\mathbf{S})$ is a matrix polynomial:

$$\mathcal{H}(\mathbf{S}) = \sum_{k=0}^{K-1} h_k \mathbf{S}^k. \tag{5}$$

where the graph filter order K can be infinite. Let the eigenvalue decomposition be $\mathbf{S} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$ where $\boldsymbol{\Lambda}$ is a diagonal matrix with eigenvalues $|\lambda_1| \leq |\lambda_2| \leq \cdots \leq |\lambda_{|\mathcal{V}|}|$ and $\mathbf{U}$ be the eigenvector matrix that is unitary since the GSO $\mathbf{S}$ is symmetric.[3] The Graph Fourier Transform (GFT) basis is $\mathbf{U}$, the GFT of a graph signal is $\tilde{\boldsymbol{x}} = \mathbf{U}^\top \boldsymbol{x}$ and the eigenvalues $\lambda_\ell, \ell = 1, \ldots, |\mathcal{V}|$ are the *graph frequencies*. From (5) it follows that:

$$\mathcal{H}(\mathbf{S}) = \mathbf{U}\left(\sum_{k=0}^{K-1} h_k \boldsymbol{\Lambda}^k\right)\mathbf{U}^\top. \tag{6}$$

The matrix $\sum_{k=0}^{K-1} h_k \boldsymbol{\Lambda}^k$ is a diagonal, with i^{th} entry $\tilde{h}(\lambda_i) \triangleq \sum_{k=0}^{K-1} h_k \lambda_i^k$. Hence, $\tilde{\boldsymbol{h}} = [\tilde{h}(\lambda_1), \ldots, \tilde{h}(\lambda_{|\mathcal{V}|})]$ is the transfer function for graph filters, and in the GFT domain, and the graph filter output corresponds to an element by element multiplication of the graph filter input, i.e.:

$$\boldsymbol{w} = \mathcal{H}(\mathbf{S})\boldsymbol{x} \iff \tilde{\boldsymbol{w}} = \tilde{\boldsymbol{h}} \odot \tilde{\boldsymbol{x}}, \tag{7}$$

where and $\odot$ represents the element by element (Hadamard) vector product. To process time series of graph signals $\{\boldsymbol{x}_t\}_{t\geq 0}$ with samples that are not i.i.d., graph temporal filters models are more appropriate:

$$\boldsymbol{w}_t = \sum_{\tau=0}^{t} \mathcal{H}_{t-\tau}(\mathbf{S})\boldsymbol{x}_\tau \qquad \mathcal{H}_t(\mathbf{S}) = \sum_{k=0}^{K-1} h_{k,t}\mathbf{S}^k, \tag{8}$$

and for their analysis we can harness DSP tools, defining a combined GFT and $z-$transform:

$$\mathbf{X}(z) = \sum_{t=0}^{K_t-1} \boldsymbol{x}_t z^{-t}, \quad \tilde{\mathbf{X}}(z) = \mathbf{U}^\top \mathbf{X}(z), \tag{9}$$

where K_t is the length of the graph signal time series. In particular, considering a filter of order K_t, we use $\mathbf{S} \otimes z$ ($\otimes$ is tensor product) as the graph temporal GSO:

$$\mathcal{H}(\mathbf{S} \otimes z) = \sum_{k=0}^{K-1} H_k(z)\mathbf{S}^k, \quad H_k(z) = \sum_{t=0}^{K_t-1} h_{k,t} z^{-t} \tag{10}$$

[3]Note that the graph frequencies of complex GSO are denoted by $|\lambda_n|$, which tend to be unique [4, 33].

i.e. $H_k(z)$ is the $z-$transform of the filter coefficients $h_{k,t}$. In the z-domain, the input-output relationship is expressed as:

$$\mathbf{W}(z) = \mathcal{H}(\mathbf{S} \otimes z)\mathbf{X}(z). \tag{11}$$

The graph-temporal transfer function and input-output relationship in the joint GFT-z-domain are:

$$\mathbb{H}(\boldsymbol{\Lambda}, z) = \sum_{t=0}^{K_t-1}\sum_{k=0}^{K-1} h_{k,t}\boldsymbol{\Lambda}^k z^{-t}, \quad \tilde{\mathbf{W}}(z) = \mathbb{H}(\boldsymbol{\Lambda}, z)\tilde{\mathbf{X}}(z), \tag{12}$$

where $\mathbb{H}(\boldsymbol{\Lambda}, z)$ is a diagonal matrix and $\tilde{\mathbf{X}}(z) = \mathbf{U}^\top \mathbf{X}(z)$, which is again an element by element multiplication since $\mathbb{H}(\boldsymbol{\Lambda}, z)$ is a diagonal matrix. In a graph-convolutional neural networks (GCN), the coefficients $h_{k,t}$ are the trainable parameter [8]. The subtle differences between complex and real GSP have been discussed in [34].

B. Complex-valued Graph Convolution Neural Network

The graph neural network perceptron based on (4) is:

$$\bar{\boldsymbol{w}} = \sigma[\boldsymbol{w}] = \sigma\left[\sum_{k=0}^{K-1} h_k \mathbf{S}^k \boldsymbol{x}\right] \tag{13}$$

where $\boldsymbol{x} \in \mathbb{C}^{|\mathcal{V}|}$, $h_k \in \mathbb{C}$, $\mathbf{S}^k \in \mathbb{C}^{|\mathcal{V}|\times|\mathcal{V}|}$ and $\boldsymbol{w} \in \mathbb{C}^{|\mathcal{V}|}$ are the complex values. Since, $\sigma(\cdot)$ takes as input complex values, there is significant flexibility in defining this operator in the complex plane. In the following, we refer to Complex ReLU (namely CReLU) as the simple complex activation that applies separate ReLUs on both of the real and the imaginary part of a neuron, i.e:

$$\mathrm{CReLU}(\boldsymbol{w}) = \mathrm{ReLU}(\Re(\boldsymbol{w})) + \mathrm{j}\,\mathrm{ReLU}(\Im(\boldsymbol{w})). \tag{14}$$

This is a popular choice because the CReLU satisfies the Cauchy-Riemann equations if both the real and imaginary parts are either strictly positive or strictly negative [2]. Empirically, we found this choice to be preferable to other options proposed in the literature.

Spatio-Temporal GCN are a special case of *multiple features GCN*. Specifically, let $\mathbf{X} = [\boldsymbol{x}^1, \cdots, \boldsymbol{x}^F]$ and let us refer to the multiple channel outputs as $\mathbf{W} = [\boldsymbol{w}^1, \cdots, \boldsymbol{w}^G]$, where F is the number of input features and G is the number of output channels. A layer of multiple features GCN operates as follows:

$$\bar{\mathbf{W}} = \sigma[\mathbf{W}] = \sigma\left[\sum_{k=0}^{K-1} \mathbf{S}^k \times \mathbf{X} \times \mathbf{H}_k\right] = \mathrm{CReLU}(\mathbf{H} *_{\mathcal{G}} \mathbf{X}), \tag{15}$$

where these matrices include $G \times F$ coefficient matrix $\mathbf{H}_k$ with entries $[\mathbf{H}_k]_{fg} = h_k^{fg}$, and $\mathbf{H}*_{\mathcal{G}}$ defines the notion of graph convolution operator based on the concept of spectral graph convolution.

1) Discussion about Cplx-STGCN vs Real-STGCN: Note that using real-GCN in lieu of complex GCN reduces significantly the number of trainable parameters. Specifically, in terms of Cplx-GCN, one way of mixing and separating the real and imaginary variables is

$$\begin{bmatrix}\Re(\boldsymbol{w})\\ \Im(\boldsymbol{w})\end{bmatrix} = \sum_{k=0}^{K-1} h_k \begin{bmatrix}\Re(\mathbf{S}) & -\Im(\mathbf{S})\\ \Im(\mathbf{S}) & \Re(\mathbf{S})\end{bmatrix}^k \begin{bmatrix}\Re(\boldsymbol{x})\\ \Im(\boldsymbol{x})\end{bmatrix} \tag{16}$$

using an h_k which is a real scalar in the decoupled model. This removes the imaginary part of h_k, reducing the neural network function approximation capability. This is why, when such GCN methods are applied to voltage phasor signals, the resulting trained models under-perform the complex ones in inference and control tasks.

IV. Analysis of the Cplx-GCN sensitivity

Particularly for power systems, it is quite common to incur in sparse system changes, due to switching or changes of line impedance. It is, therefore, of interest to understand how sensitive is the response of the Cplx-GCN to changes in the parameters. For the case where the changes in the GSO are known it is to study how parameters trained on a different GSO will respond. We refer to this as the *transfer learning* error. Next we provide insights on the impact of perturbations in the GSO on the end-to-end Cplx-GCN mapping. We improve substantially the results of [18] which are exclusively for real GNN, do not consider the end to end distortion, and also rely on restrictive assumptions about the structure of the perturbation that we could not justify in our practical setting.

In the following we denote by $\sigma_{\max}(\mathbf{A})$ its largest singular value of matrix $\mathbf{A}$. We can prove the following bound:

Theorem 1 *Consider graph filter $\boldsymbol{h} = [h_0, \cdots, h_K]$ along with shift operator $\mathbf{S}$ having $|\mathcal{V}|$ nodes. Let $\mathbf{E} \in \mathbb{C}^{|\mathcal{V}|\times|\mathcal{V}|}$ denote the matrix perturbation with $\|\mathbf{E}\| \leq \epsilon$, and $\hat{\mathbf{S}} = \mathbf{S} + \mathbf{E}$. Let us denote by $\hat{\boldsymbol{h}} = [\hat{h}_0, \cdots, \hat{h}_K]$ the graph filter parameters obtained training the network using $\hat{\mathbf{S}}$ as GSO. Let $\hat{\mathcal{H}}(\hat{\mathbf{S}}) = \sum_{k=0}^{K} \hat{h}_k \hat{\mathbf{S}}^k$ and the cplx-GCN layer with the the original filter coefficients obtained training with GSO $\mathbf{S}$, albeit using the perturbed GSO, be $\mathcal{H}(\hat{\mathbf{S}}) = \sum_{k=0}^{K} h_k \hat{\mathbf{S}}^k$. Let us also define:*

$$\gamma_1 \triangleq \max\left(1, (\sigma_{\max}(\mathbf{S}) + \epsilon)^K\right) \tag{17}$$

The following bound holds:

$$\sigma_{\max}(\hat{\mathcal{H}}(\hat{\mathbf{S}}) - \mathcal{H}(\hat{\mathbf{S}})) \leq \gamma_1 \|\hat{\boldsymbol{h}} - \boldsymbol{h}\|_1. \tag{18}$$

Proof 1 *The proof is in Appendix A.*

Theorem 2 *Let $\mathbf{E} \in \mathbb{C}^{|\mathcal{V}|\times|\mathcal{V}|}$ be the matrix perturbation with $\|\mathbf{E}\| \leq \epsilon$, and $\hat{\mathbf{S}} = \mathbf{S} + \mathbf{E}$ and*

$$\gamma_2 \triangleq \max_{1\leq k\leq K} |h_k|(1 + \sigma_{\max}(\mathbf{S}))^K. \tag{19}$$

Assume that the cplx-GCN layer used is $\mathcal{H}(\hat{\mathbf{S}}) = \sum_{k=0}^{K} h_k \hat{\mathbf{S}}^k$ where the coefficients are the same as those obtained by training in the original cplx-GCN layer is defined as $\mathcal{H}(\mathbf{S}) = \sum_{k=0}^{K} h_k \mathbf{S}^k$. Then, the following bound holds:

$$\sigma_{\max}(\mathcal{H}(\hat{\mathbf{S}}) - \mathcal{H}(\mathbf{S})) \leq \gamma_2 \frac{\epsilon(1-\epsilon^K)}{1-\epsilon}. \tag{20}$$

Proof 2 *The proof is in Appendix B.*

This theorem clearly shows that for a small perturbation in the GSO one should get a similar response to the parameters, which suggests that for small GSO perturbations is reasonable to use the same parameters and transfer the learning done to the new case.

Next we bound the difference between the two outputs of the retrained network with the perturbed GSO and the original network, in other words the bound on the norm of the output difference, when the GCN perturbation consists of the perturbations of both parameters h_k and GSO.

Corollary 1 (The Bound of Cplx-GCN Perturbation) *Let the retrained cplx-GCN layer be $\hat{\mathcal{H}}(\hat{\mathbf{S}}) = \sum_{k=0}^{K} \hat{h}_k \hat{\mathbf{S}}^k$ and the original one be $\mathcal{H}(\mathbf{S}) = \sum_{k=0}^{K} h_k \mathbf{S}^k$ with the new GSO $\hat{\mathbf{S}}$. Then:*

$$\begin{aligned}\|\big(\hat{\mathcal{H}}(\hat{\mathbf{S}}) - \mathcal{H}(\mathbf{S})\big)\boldsymbol{x}\| \leq \\ \left(\gamma_1 \|\hat{\boldsymbol{h}} - \boldsymbol{h}\|_1 + \gamma_2 \frac{\epsilon(1-\epsilon^K)}{1-\epsilon}\right)\|\boldsymbol{x}\|\end{aligned} \tag{21}$$

where the parameters in the right-hand side were defined in Theorem 1 and 2.

The proof is obvious because of the triangle inequality.

Remark 1 *If the GSO $\mathbf{S}$ is normalized by $\sigma_{\max}(\mathbf{S})$, we can further bound the result in* (18) *as follows:*

$$\gamma_1 \|\hat{\boldsymbol{h}} - \boldsymbol{h}\|_1 \leq (1+\epsilon)^K \|\hat{\boldsymbol{h}} - \boldsymbol{h}\|_1 \tag{22}$$

γ_1 amplifies the sensitivity exponentially K while the effect of the parameters difference increases linearly with K. When $K \to \infty$, we have

$$\lim_{K\to\infty} (1+\epsilon)^K \|\hat{\boldsymbol{h}} - \boldsymbol{h}\|_1 = e^{\epsilon K} \|\hat{\boldsymbol{h}} - \boldsymbol{h}\|_1 \tag{23}$$

Moreover, if ϵ is approaching 0, i.e., $\epsilon \to \frac{1}{K}$, the error can be further bounded by

$$\lim_{K\to\infty, \epsilon\to\frac{1}{K}} (1+\epsilon)^K \|\hat{\boldsymbol{h}} - \boldsymbol{h}\|_1 = e \|\hat{\boldsymbol{h}} - \boldsymbol{h}\|_1 \tag{24}$$

Remark 2 *The interpretation of Theorem 2 is more straightforward. Its dependence on ϵ is clear, and it is also clear that it exponentially increases with K with a rate $(1 + \sigma_{\max}(\boldsymbol{S}))$. So, when using an incorrect GSO in the network one would want to make sure that $\epsilon \ll (1+\sigma_{\max}(\mathbf{S}))^{-K}$. Here, $\sigma_{\max}(\mathbf{S})$ and K are clearly negatively impacting the sensitivity.*

A. Spatiotemporal Graph Convolutional Neural Network

Power systems are dynamic systems with time-varying voltage phasors. In order to fuse features from both spatial and temporal domains, we will consider the following two graph neural network architectures, namely Conv1D Graph convolutional neural networks. Although RNN-based models become widespread in time-series analysis, RNN for power systems still suffers from time-consuming iterations, complex gate mechanisms, and slow response to dynamic changes. CNNs, on the other hand, allow for fast training, have a simpler structures, and no dependency constraints from previous steps. As shown in Fig. 1, the temporal convolutional layer contains a 1-D CNN with a width-T kernel with K_t output channels. The convolutional kernel $\mathbf{\Gamma} \in \mathbb{C}^{T\times K_t}$ is designed to map the input $\mathbf{X} \in \mathbb{C}^{|\mathcal{V}|\times T}$ into a output graph signal with C_t channels $\bar{\mathbf{X}} \in \mathbb{C}^{|\mathcal{V}|\times K_t}$. Therefore, we define the temporal convolution as,

$$\bar{\mathbf{X}} = \mathbf{\Gamma} *_{\mathcal{T}} \mathbf{X}, \tag{25}$$

where each column of $[\bar{\mathbf{X}}]_\tau$ is defined as $\bar{\boldsymbol{x}}_\tau, \tau = 0, 1, \cdots, K_t - 1$. After the temporal convolutional layer, we are ready to put $\bar{\mathbf{X}}$ into the spatial layer. Based on (12), we can design the following transfer functions and neuron:

$$\begin{aligned} \mathbb{H}(\mathbf{S}, z) &= \sum_{t=0}^{K_t-1} \sum_{k=0}^{K-1} h_{k,t} \mathbf{S}^k z^{-t}, \\ \bar{\boldsymbol{w}}_t = \sigma[\boldsymbol{w}_t] &= \sigma \left[\sum_{k=0}^{K-1} \sum_{\tau=0}^{K_t-1} h_{k,\tau} \mathbf{S}^k \boldsymbol{x}_{t-\tau} \right]. \end{aligned} \tag{26}$$

Accordingly, the graph signal $\bar{\boldsymbol{w}}_t$ from the spatial feature extraction layer (see Fig. 1) is:

$$\bar{\boldsymbol{w}}_t = \mathrm{CReLU} \left[\sum_{k=0}^{K-1} \sum_{\tau=0}^{K_t-1} h_{k,\tau} \mathbf{S}^k \bar{\boldsymbol{x}}_{t-\tau} \right], \tag{27}$$

By combining the temporal and spatial convolutions at each layer, the multiple output channels of the Cplx-STGCN layer ($\ell = 1$) are expressed as

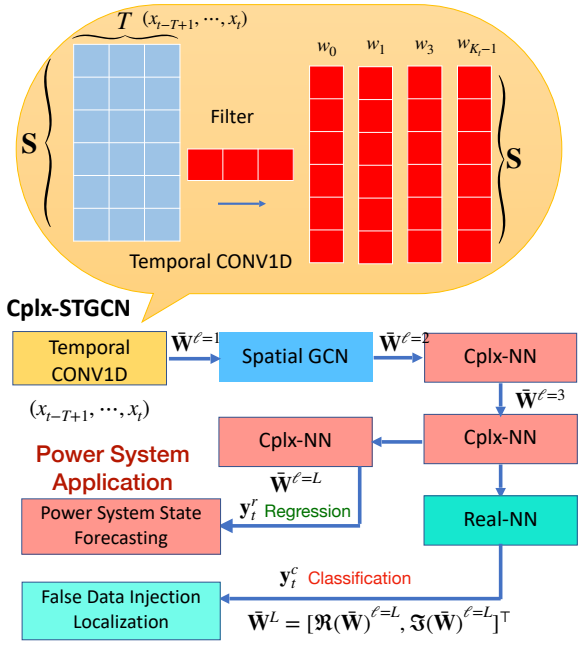


Fig. 1: The Architecture of Cplx-STGCN.

The following hidden layers $\ell \in \{1, \cdots, L-1\}$ are the complex-valued fully connected neural network:

$$\bar{\mathbf{W}}_{t,\ell+1} = \mathrm{CReLU}\left(\Theta_\ell^{cplx} * \bar{\mathbf{W}}_{t,\ell}\right), \quad L - 1 \geq \ell \geq 1. \tag{29}$$

For the output layer L, we transform the complex tensor $\bar{\mathbf{W}}_{t,L}$ into a real tensor, and then map it to the labels (or regression targets):

$$\begin{aligned} \text{regression: } \boldsymbol{y}^r &= \tanh\left(\Theta_L^{cplx} * \bar{\mathbf{W}}_{t,L}\right), \\ \text{classification: } \boldsymbol{y}^c &= \text{sigmoid}\left(\Theta_L^{re} * \begin{bmatrix} \Re(\bar{\mathbf{W}}_{t,L}) \\ \Im(\bar{\mathbf{W}}_{t,L}) \end{bmatrix}\right). \end{aligned} \tag{30}$$

where $\boldsymbol{y}^r$ and $\boldsymbol{y}^c$ denote the complex regression targets and the real classification labels, respectively. Besides, Θ_L^{cplx} and Θ_L^{re} denote the complex and real trainable matrix. Then, we define the multi-layer Cplx-STGCN learning function as:

$$\begin{aligned} \text{regression: } \boldsymbol{y}_t^r &= \Phi^r(\mathbf{X}_t, \mathbf{S}, \theta^r), \\ \text{classification: } \boldsymbol{y}_t^c &= \Phi^c(\mathbf{X}_t, \mathbf{S}, \theta^c), \end{aligned} \tag{31}$$

where $\theta^r \triangleq \{(\Theta_\ell^{cplx}, \mathbf{H}, \mathbf{\Gamma}) | \forall \ell = 1, \cdots, L\}$ and $\theta^c \triangleq \{(\Theta_\ell^{cplx}, \Theta_L^{real}, \mathbf{H}, \mathbf{\Gamma}) | \forall \ell = 1, \cdots, L-1\}$ represent the trainable parameters and $\mathbf{X}_t = [\boldsymbol{x}_{t-T+1}, \cdots, \boldsymbol{x}_t]$. Here, we have omitted the bias term to unburden the notation, but they are present in the trainable model we use.

In the following, we further investigate how the multilayer neural networks propagate the error due to the changes of parameter and GSO $\mathbf{S}$.

Lemma 1 *Assume a neural network consists of one Cplx-GCN layer and one Cplx-FNN layer with trainable parameters Θ^{cplx}, denoted by $\boldsymbol{y} = \Phi(\boldsymbol{x}, \mathbf{S}, \theta)$. The retrained neural network has the new GSO $\hat{\mathbf{S}}$ and the new parameters $\hat{\theta}$, denoted by $\hat{\boldsymbol{y}} = \Phi(\boldsymbol{x}, \hat{\mathbf{S}}, \hat{\theta})$. We define the perturbation of the Cplx-FNN layer is $\left\|\Theta^{cplx} - \hat{\Theta}^{cplx}\right\|_2 \leq \delta_{\mathbf{w}}$. Then, the distance between $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ is bounded by:*

$$\|\boldsymbol{y} - \hat{\boldsymbol{y}}\| \leq \Big(\overbrace{\delta_{\mathbf{w}} * \Psi_1 + \sigma_{\max}(\Theta^{cplx}) * \Psi_2}^{\triangleq \Delta_1} \Big) \|\boldsymbol{x}\| \tag{32}$$

where Ψ_1 and Ψ_2 are given by

$$\begin{aligned} \Psi_1 &= \gamma_1 \|\hat{\boldsymbol{h}}\|_1 \\ \Psi_2 &= \left[\gamma_1 \left\| \hat{\boldsymbol{h}} - \boldsymbol{h} \right\|_1 + \gamma_2 \frac{\epsilon(1 - \epsilon^K)}{1 - \epsilon} \right] \end{aligned} \tag{33}$$

Proof 3 *The proof is in Appendix C.*

Corollary 2 *We generalize the bound into the multilayer neural networks, including one Cplx-GCN feature extraction layer and L cplx-FNNs as*

$$\begin{aligned} &\|\boldsymbol{y} - \hat{\boldsymbol{y}}\| \leq \Delta_L, \quad \Delta_L = \sigma_{\max}(\Theta_L^{cplx}) \Delta_{L-1} \\ &+ \delta_{\mathbf{w}_L} \prod_{\ell=1}^{L-1} \sigma_{\max}(\hat{\Theta}_\ell^{cplx}) \Psi_1. \end{aligned} \tag{34}$$

where Δ_1 is defined in (32).

The proof is obvious according to the norm triangle inequality so that we omit the proof. From Corollary 2, we can observe that if the retrained neural networks have small changes to make $\delta_{\mathbf{w}_L} \approx 0$, we could find the dominant part, i.e., $\Delta_L \approx \sigma_{\max}(\Theta_L^{cplx}) \Delta_{L-1}$. Therefore, the error of the GCN perturbation is propagated by the largest singular values of original cplx-FNN weight matrices.

V. Applications of Cplx-STGCN

A. Power System State Estimation and Forecasting

Measurements in power systems are relatively sparse. In this subsection, we propose a Power Systems State Estimation (PSSE) algorithm that can use limited measurements to estimate the current and future state at all buses. Let $\boldsymbol{x}_{\mathcal{A}}$ (the time index t is ignored for simplicity) be the vector of measurements in the subset of buses $\mathcal{A} \subset \mathcal{V}$ that have

sensors. Let the GFT basis corresponding to the dominant k graph frequencies in the voltage phasor GFT spectrum $\mathbf{U}_{\mathcal{K}}$. Because of Ohm's law, this set corresponds to the lowest graph frequencies [4] and the best subset of measurement buses $\mathcal{A}$ is one-to-one with the subset of rows of $\mathbf{U}_{\mathcal{K}}$ with minimum correlation. Let $\mathcal{F}_{\mathcal{A}}$ be the so called *vertex limiting operator* i.e. the matrix such that $\mathcal{F}_{\mathcal{A}} = \mathcal{Q}_{\mathcal{A}}\mathcal{Q}_{\mathcal{A}}^{\top}$, where $\mathcal{Q}_{\mathcal{A}}$ has columns that are the coordinate vectors pointing to each vertex/node in $\mathcal{A}$, so that $\boldsymbol{x}_{\mathcal{A}} = \mathcal{Q}_{\mathcal{A}}\boldsymbol{x}$. Mathematically, the optimal placement can be sought by maximizing the smallest singular value, $\max_{\mathcal{F}_{\mathcal{A}}} \varpi_{\min}(\mathcal{F}_{\mathcal{A}}\mathbf{U}_{\mathcal{K}})$, of the matrix $\mathcal{F}_{\mathcal{A}}\mathbf{U}_{\mathcal{K}}$. Such choice amounts to the selection of rows of $\mathbf{U}_{\mathcal{K}}$ that are as uncorrelated as possible, because the resulting matrix $\mathcal{F}_{\mathcal{A}}\mathbf{U}_{\mathcal{K}}$ has the highest condition number [35].

After choosing, using the aforementioned method, the best location for measuring the voltage phasors[4] $\mathcal{A}$ the available measurements for the vector observation $\boldsymbol{z}_t$. With $\mathcal{A}$ denoting the set of available measurement, we use $\mathcal{U}$ to denote the set of unavailable ones. Therefore, (2) can be written as:

$$\underbrace{\begin{bmatrix} \hat{\boldsymbol{i}}_{\mathcal{A}} \\ \hat{\boldsymbol{v}}_{\mathcal{A}} \end{bmatrix}}_{\boldsymbol{z}_t} = \underbrace{\begin{bmatrix} \boldsymbol{Y}_{\mathcal{A}\mathcal{A}} & \boldsymbol{Y}_{\mathcal{A}\mathcal{U}} \\ \mathbb{I}_{|\mathcal{A}|} & \mathbf{0} \end{bmatrix}}_{\boldsymbol{H}} \underbrace{\begin{bmatrix} \boldsymbol{v}_{\mathcal{A}} \\ \boldsymbol{v}_{\mathcal{U}} \end{bmatrix}}_{\boldsymbol{x}_t} + \boldsymbol{\varepsilon}_t, \tag{35}$$

where $\boldsymbol{\varepsilon}_t$ is a vector of measurement noise. The voltage phasor forecasting is a typical time-series prediction problem, i.e. predicting the most likely voltage phasor measurements in the next H time steps given the previous T sub-sampled measurement $[\boldsymbol{x}_t]_{\mathcal{A}}$ as

$$\boldsymbol{x}^*_{t+H} = \underset{\boldsymbol{x}_{t+H}}{\arg\max} \log P\left(\boldsymbol{x}_{t+H} \mid [\boldsymbol{x}_{t-T+1}]_{\mathcal{A}}, \ldots, [\boldsymbol{x}_t]_{\mathcal{A}}\right), \tag{36}$$

where $[\boldsymbol{x}_t]_{\mathcal{A}} \in \mathbb{C}^{|\mathcal{A}|}$ is an observation vector of $|\mathcal{A}|$ measurements at time step t, each element of which records historical observation for a bus.

1) Methodology: The first step of the algorithm is to recover the voltage phasors $\boldsymbol{x}_t$ from $\boldsymbol{z}_t = \left[\hat{\boldsymbol{i}}_{\mathcal{A}}, \hat{\boldsymbol{v}}_{\mathcal{A}}\right]^{\top}$ by solving the regularized least square problem:

$$\min_{\boldsymbol{x}_t} \|\boldsymbol{z}_t - \boldsymbol{H}\boldsymbol{x}_t\|_2^2 + \mu_1(\boldsymbol{x}_t^H \mathbf{S} \boldsymbol{x}_t) \tag{37}$$

where μ_1 is positive. The closed-form solution of (37) is:

$$\hat{\boldsymbol{x}}_t = \left(\boldsymbol{H}^H\boldsymbol{H} + \mu_1\mathbf{S}\right)^{\dagger}\boldsymbol{H}^H\boldsymbol{z}_t, \tag{38}$$

where $\hat{\boldsymbol{x}}_t$ is the estimated voltage phasor and $(\cdot)^{\dagger}$ denotes the pseudo-inverse. The algorithm step are as follows:

1) We collect T historical measurements $\boldsymbol{z}_{t-T+1}, \cdots, \boldsymbol{z}_t$.
2) We utilize (38) to obtain the estimated full observations $\hat{\mathbf{X}} = [\hat{\boldsymbol{x}}_{t-T+1}, \ldots, \hat{\boldsymbol{x}}_t]$.
3) The Cplx-STGCN loss function for the voltage phasor prediction is written as

$$\mathcal{L}(\Phi, \theta) = \sum_t \Big\{ \left\|\boldsymbol{y}_t^r - \boldsymbol{x}_{t+H}\right\|^2 + \mu_2 \left\|\hat{\boldsymbol{v}}_{t+H,\mathcal{A}} \circ \hat{\boldsymbol{i}}^*_{t+H,\mathcal{A}} - \left[\boldsymbol{y}_t^r \circ (\mathbf{S}\boldsymbol{y}_t^r)^*\right]_{\mathcal{A}}\right\|^2 \Big\}, \tag{39}$$

where $\boldsymbol{x}_{t+H}$ is the ground truth voltage phasor in the next H time step and $\boldsymbol{y}_t^r = \Phi^r(\hat{\mathbf{X}}_t, \mathbf{S}, \theta)$ in (31) is the predicted target to approximate the ground-truth regression target $(\boldsymbol{x}_{t+H})$, and the regularization term in (39) favors voltage phasor forecasts that minimize the sum of the absolute value of the apparent power injections. Note that $H = 0$ is the voltage phasor estimation, and $H \geq 1$ is the voltage phasor forecasting.

After training, we use $\Phi^{r*}(\hat{\mathbf{X}}_t, \mathbf{S}, \theta^*)$ to forecast the outputs $\boldsymbol{x}_{t+H}$ from the inputs $[\boldsymbol{x}_{t-T+1}]_{\mathcal{A}}, \ldots, [\boldsymbol{x}_t]_{\mathcal{A}}$ that are obtained from the observations at the corresponding times.

[4]These measurements can be collected by Phasor Measurement Units.

B. False Data Detection and Localization

The conventional task of FDI attack detection is a binary hypothesis testing problem where the null-hypothesis is *no false data are present* and the positive one is that *some data are compromised*. The localization of the false measurement (FDI localization problem), is the one of interest in this subsection. Such problem amounts to classifying each measurements into two categories (false data or not) and, thus, is a multi-label classification problem.

In a stealth attack [4], the attacker manipulates both current and voltage phasor measurements on the subset buses $\mathcal{C}$ by introducing a perturbation:

$$\delta\boldsymbol{x}_t^{\top} = \begin{bmatrix} \delta\boldsymbol{x}_{\mathcal{C}}^{\top} & \mathbf{0}_{|\mathcal{P}|+|\mathcal{U}|}^{\top} \end{bmatrix}, \text{ such that } \boldsymbol{Y}_{\mathcal{P}\mathcal{C}}\delta\boldsymbol{x}_{\mathcal{C}} = \mathbf{0}, \mathcal{C} \subset \mathcal{A} \tag{40}$$

where $\mathcal{P}$ is the set of honest nodes. This requires special conditions and placement, since $\boldsymbol{Y}_{\mathcal{P}\mathcal{C}}$ is tall and does not have full column-rank for a sufficient number of attacker $\mathcal{C}$. Therefore, the received data $\boldsymbol{z}_t$ with FDI attack have the structure:

$$\boldsymbol{z}_t = \boldsymbol{H}(\boldsymbol{x}_t + \delta\boldsymbol{x}_t) + \boldsymbol{\varepsilon}_t. \tag{41}$$

1) Methodology: Then, the algorithm for FDI localization is summarized as

1) We obtain T historical measurements $\boldsymbol{z}_{t-T+1}, \cdots, \boldsymbol{z}_t$ with FDI attacks based on (41).
2) We construct the ground-truth label vector $\boldsymbol{y}'_t = \text{logit}(\delta\boldsymbol{x}_t)$, where $\text{logit}(\cdot)$ is an indicator function that $[\boldsymbol{y'}_t]_i = 1$ if $[\delta\boldsymbol{x}_t]_i \neq 0$, otherwise $[\boldsymbol{y}'_t]_i = 0$. Note that $\mathcal{C} \subset \mathcal{A}$ is the set of randomly sampled buses with the fixed number.
3) We utilize (38) to obtain the estimated voltage phasors $\hat{\boldsymbol{x}}_{t-T+1}, \cdots, \hat{\boldsymbol{x}}_t$.
4) The loss function of the Cplx-STGCN function for voltage phasor prediction is written as

$$\mathcal{L}(\Phi^c, \theta^c) = \sum_t \left\|\Phi^c(\hat{\mathbf{X}}_t, \mathbf{S}, \theta^c) - \boldsymbol{y}'_t\right\|^2. \tag{42}$$

With the trained $\Phi^{c*}(\hat{\mathbf{X}}_t, \mathbf{S}, \theta^c)$, we could predict the multiple labels $\boldsymbol{y}_t^c = \Phi^{c*}(\hat{\mathbf{X}}_t, \mathbf{S}, \theta^c)$ when $\boldsymbol{z}_{t-T+1}, \cdots, \boldsymbol{z}_t$ are observed.

VI. Numerical Experiments

The numerical results in this section are obtained from the IEEE 118-bus case with 118 nodes and 186 edges [36]. This system includes 54 generators, 118 buses (nodes) and 186

edges, whose system parameters can be found in Matpower [36]. We collect realistic demand profiles from the Texas grid and use Matpower [36] to compute the voltage phasors. In all simulations, we repeat the training and testing 8 times to report the average values.

TABLE I: Sensor Measurement Installed Buses

Systems	Bus Name
ieee 118-bus with PMUs	14, 117, 72, 86, 43, 67, 99, 87, 16, 33, 112, 28, 98, 111, 53, 97, 42, 107, 48, 22, 46, 13, 24, 101, 44, 73, 109, 29, 20, 91 26, 84, 10, 1, 52, 57, 76, 115, 39, 74, 104, 93, 79, 35, 6, 18, 88, 60, 116, 55, 58, 68, 64, 7, 50, 103, 75, 78, 83, 69.

Cplx-STGCN Setting: The architecture of the Cplx-STGCN for PSSF and FDIA locational detection includes the Cplx-STGCN layer, the two-layer Cplx-NNs and one real-valued output layer. In the Cplx-STGCN layer, the Cplx-CNN (temporal convolution) output channel is 10 and the Cplx-GCN (spatial convolution) output channel is 10. Other Cplx-NNs have 512 neurons per layer. The order of GSO K in the Cplx-GCN is 5.

Baseline Setting: The baseline algorithms for both applications include fully-connected NN (**FNN**) that has 4 layers with 512 neurons each layers, complex-valued fully-connected NN (**CplxFNN**) that has 4 layers with 512 neurons each layers, and **GCN1st** that has the first layer GNN [37], and 3 layers of fully connected NNs. Specific algorithms for PSSF include **RNN** [28] that incorporates a lot of measurements (i.e. voltage magnitudes, active and reactive power injections) and **GRNN** [29] that combines a GNN1st layer and a LSTM layer together to capture the spatio-temporal correlations. Likewise, the state-of-art algorithms for false data detection and localization include **ChebyGCN** [30] that uses the absolute values of the admittance matrix and consider active and reactive powers as inputs, **CNN** [31] that takes both line and bus measurements[5], and **LSTM** [32] that considers voltage phasors as inputs, and **GSP** algorithm [4].

Sensor Placement: We take the eigendecomposition of $\boldsymbol{Y}$ and choose $|\mathcal{K}| = 40$ graph frequency components. Then, we choose the number of sensor placements $|\mathcal{A}| = 60$ and place them so as to maximize $\max_{\mathcal{F}_\mathcal{A}} \varpi_{\min}(\mathcal{F}_\mathcal{A}\mathbf{U}_\mathcal{K})$. The resulting of sensor placement is shown in Table I. Besides, through numerous simulations for the hyperparameter tuning, we choose $\mu_1 = 1e-6$ and $\mu_2 = 1e-4$ for all benchmarks.

A. Power System State Estimation and Forecasting

1) Power System State Forecasting Setting: All the tests use 10 hours as the historical time window, a.k.a. 10 observed data points $T = 10$ are used to forecast voltage phasors in the next 1, 2, 3, 4, and 5 hours ($H = 1, 2, 3, 4, 5$). If $H = 0$, it is a PSSE problem that estimates the complete voltage phasors $\boldsymbol{x}_t$ given $[\boldsymbol{x}_{t-T+1}]_\mathcal{A}, \ldots, [\boldsymbol{x}_t]_\mathcal{A}$.

2) Results: Figs. 2(a) and 2(c) show the results of Cplx-STGCN and various baselines on the IEEE 118-bus system experiment described above. The Cplx-STGCN achieves the best performance. In particular, $H = 0$ is the PSSE problem, and the MSE of (38) for estimation is 0.0002371. While this is a respectable outcome, the supervised Cplx-STGCN has much smaller error, i.e. 0.00003517. Another observation is that the voltage phasors predicted by Cplx-STGCN could approximate the OPF results with much smaller MAPE, e.g. 0.5359% at $H = 4$, compared with other methods, e.g. 2.0875% of GRNN and 4.3227% of FNN. We illustrate two examples in Fig. 4 to show the ground-truth fully-observed voltage magnitudes and phase angles with the predicted ones with $H = 0, 1$, which shows the predicted voltage phasors are very close to the ground-truth.

3) Transferability of Cplx-STGCN Regarding Topology Changes: In this section, we validate the transferability of the proposed Cplx-STGCN against the topology changes. Retraining a new model based on the new topology is time-consuming. To handle this problem, we keep the trained parameters unchanged and modify the GSO of Cplx-STGCN corresponding to the topology changes of power grids. In this simulation, we trip one line of power grids as the new topology. The results are shown in Fig. 2(b) and 2(d), which indicates that Cplx-STGCN performs well in the new topology. However, the fully-connected neural networks do not adapt to the new topology well. This is because the GSO captures the topology changes while the fully-connected neural networks do not have this property.

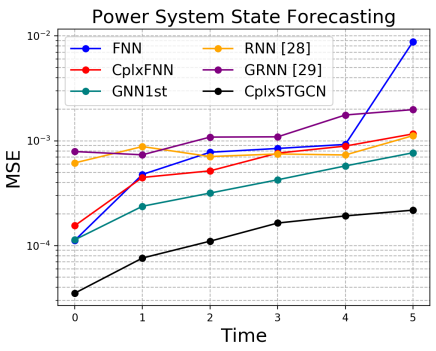
B. False Data Injection Localization

1) FDI Setting: The output of Cplx-STGCN $\boldsymbol{y}_t \in [0, 1]^{|\mathcal{A}|}$ can be classified by a discrimination threshold (i.e. 0.5) to quantify the outputs to 0 or 1. The discrimination threshold can be adjusted to increase or decrease the sensitivity to application factors. Unless specified, the discrimination threshold is set to 0.5 in this article following the common practice. Likewise, 10 observed data points $T = 10$ are used for FDI localization. The number of measurements $|\mathcal{A}|$ is 60, and the 60 binary labels $\boldsymbol{y}_t = [y_1, y_2, \cdots, y_{60}]^\top$ are converted into one label with a class size of 2^{60}. Note that, $|\mathcal{C}|$ denotes the number of buses attacked, which is chosen from 25 to 50. We also provide the accuracy, **precision**, **recall**, and F_1 **scores** with biases for the Cplx-STGCN.

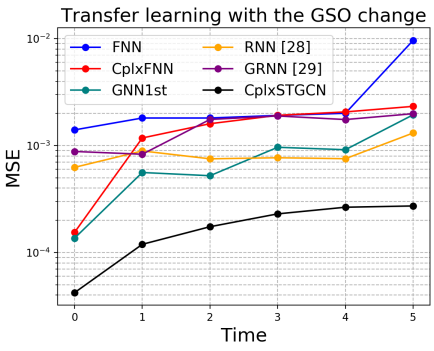
2) Results: The performance comparison for FDI localization is in Fig. 3(a). We can observe that Cplx-STGCN has much higher accuracy over other NNs and the GSP method that solves a LASSO problem to detect the false data entries. Other NNs tend to predict all ones or all zero vectors depending on whether $|\mathcal{C}|$ is large or small, respectively, while Cplx-STGCN captures the high-order spatial dependency and temporal correlation of voltage phasors, and thus achieve better performance as a result. Another observation is Cplx-STGCN has higher accuracy than LSTM [32], especially when $|\mathcal{C}|$ is large. Finally, Cplx-STGCN exhibits more stable performance with different values $|\mathcal{C}|$.

3) FDI Localization for Hybrid Dataset: In the previous subsection, we have considered the attack hypotheses that for every $\boldsymbol{z}_t, \forall t$, FDI attacks are launched on some buses $\mathcal{C}$ of $\boldsymbol{z}_t$. In this subsection, we test the proposed algorithm on the data set under both no-attack (H_0) and attack (H_1) hypotheses.

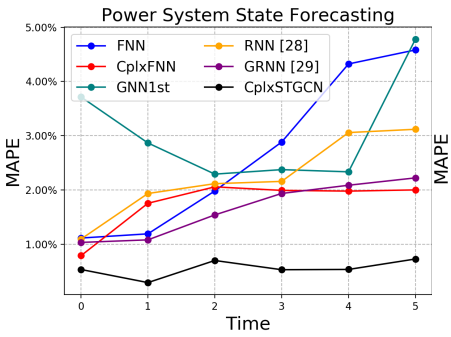
[5]For a fair comparision, we choose the same number of sensors as our algorithm



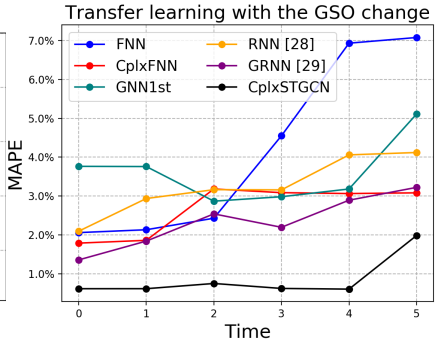
(a) MSE: PSSF for Voltage Phasors



(b) MSE: PSSF for Voltage Phasors with the GSO change

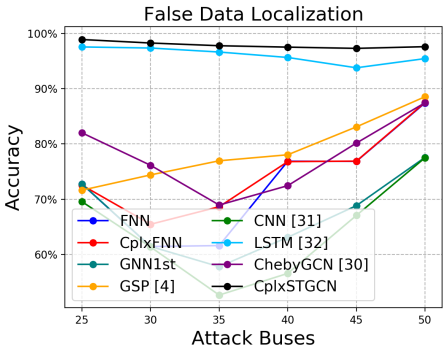


(c) MAPE: PSSF for Fuel Costs

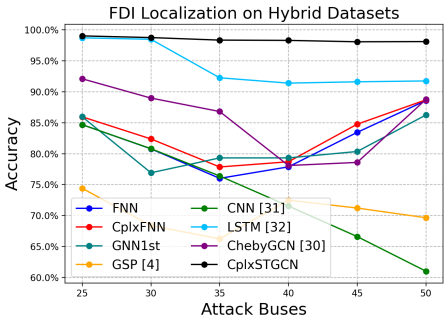


(d) MAPE: PSSF for Fuel Costs with the GSO change

Fig. 2: Power System State Estimation and Forecasting in the IEEE 118-bus system

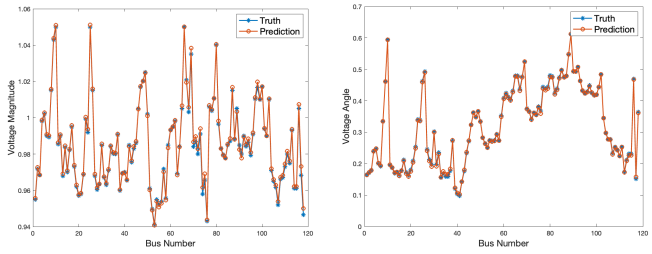


(a) Accuracy: False Data Injection Localization

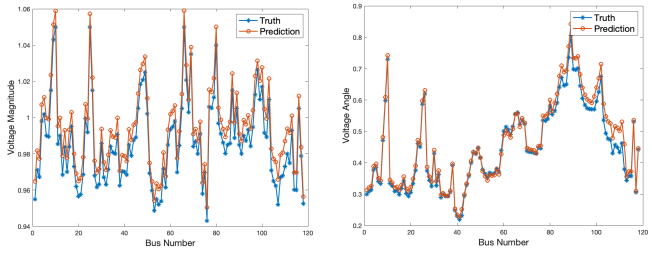


(b) Accuracy: FDI Localization for Hybrid Datasets

Fig. 3: False Data Injection Localization in the IEEE 118-bus system.



(a) Voltage Magnitudes ($H = 0$). (b) Voltage Phase Angles ($H = 0$).



(c) Voltage Magnitudes ($H = 1$). (d) Voltage Phase Angles ($H = 1$).

Fig. 4: An example of PSSE and forecasting.

Therefore, the received data $\boldsymbol{z}_t$ with FDI attack have the structure:

$$\boldsymbol{z}_t = \begin{cases} H_0 : \boldsymbol{H}(\boldsymbol{x}_t) + \boldsymbol{\varepsilon}_t \\ H_1 : \boldsymbol{H}(\boldsymbol{x}_t + \delta \boldsymbol{x}_t) + \boldsymbol{\varepsilon}_t \end{cases} . \tag{43}$$

Fig.3(b) shows the simulation results for the data set under both no-attack (H_0) and attack (H_1) hypotheses. It shows that the proposed algorithm has very high accuracy, e.g. 98.1043%, compared with other methods, e.g. 88.5806% of FNN and 91.7466% of LSTM with $|\mathcal{C}| = 50$.

TABLE II: FDI Localization Metrics

Metrics	$\|\mathcal{C}\| = 25$	$\|\mathcal{C}\| = 30$	$\|\mathcal{C}\| = 35$	$\|\mathcal{C}\| = 40$	$\|\mathcal{C}\| = 45$	$\|\mathcal{C}\| = 50$
Accuracy	98.8825%	98.2500%	97.7617%	97.5067%	97.2827%	97.5883%
Precision	96.2615%	96.0518%	96.0790%	96.1384%	96.8030%	97.5457%
Recall	99.6952%	99.5566%	99.1962%	99.8886%	98.7180%	99.0236%
F_1 Score	97.9426%	97.7680%	97.6091%	97.9752%	97.7490%	98.2778%

TABLE III: FDI Localization Metrics for Hybrid Datasets

Metrics	$\|\mathcal{C}\| = 25$	$\|\mathcal{C}\| = 30$	$\|\mathcal{C}\| = 35$	$\|\mathcal{C}\| = 40$	$\|\mathcal{C}\| = 45$	$\|\mathcal{C}\| = 50$
Accuracy	99.0230%	98.7502%	98.3413%	98.2958%	98.0729%	98.1043%
Precision	94.2437%	94.4576%	94.3624%	95.5202%	95.8494%	96.9704%
Recall	99.1154%	99.0222%	98.2522%	98.4301%	97.7849%	98.0426%
F_1 Score	96.5963%	96.6700%	96.2546%	96.9452%	96.8011%	97.4988%

4) Other Metrics of FDI Localization: In this section, we show other metrics, including precision, recall and F_1 score. Let True Negatives (TN) refer to the unattacked buses that are classified as unattacked buses. True Positives (TP) refer to the FDI attacked bus correctly predicted to be attacked. False Negatives (FN) refer to the unattacked buses that are predicted to be attacked, and False Positives (FP) refer to the attacked buses that are predicted to be unattacked. Moreover, precision is defined as the ratio of the number of TP to the total number of buses that are actually attacked. Likewise, recall (True Positive Ratio (TPR)) is defined as the ratio of the number of TP to the number of true attacked buses

$$\text{Precision} = \frac{TP}{TP + FP}, \qquad \text{Recall} = \frac{TP}{TP + FN}. \tag{44}$$

To strike a balance between the precision and recall, we define F_1-Score as the geometrical average of the precision and recall. Eq. (45) shows the F_1-Score calculation.

$$F_1\text{-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{45}$$

The results are shown in Tables II and III. It shows that Precision, Recall and F_1 Score are very high. It indicates that the false alarm rate of the Cplx-STGCN is small and for both balanced and unbalanced data sets.

VII. Conclusions

In this paper, we presented a new complex-valued spatio-temporal graph convolutional neural network architecture for the complex-valued graph signals and graph shift operator. Among the key advantages of our approach compared to traditional methods is its more informative and compact representation for complex-valued GS. We show two applications of the proposed Cplx-STGCN in power systems that have the complex-valued GSO and GS, including power system state forecasting and FDI localization. We prove that complex-valued GCNs are stable with respect to perturbations of the underlying graph support and both the transfer error and the propagation error through multiply layers are bounded. The results of the experiments attest the potential of the nascent field of geometric deep learning in the complex domain, and can spur future research in Artificial Intelligence for energy system, wireless communication and biological networks whose signals are sparse in the Fourier domain.

Appendix

A. Proof of Theorem 1

Proof: We want to characterize how much a change in $\hat{\mathbf{S}}$ changes the response of the feature extraction layer as

$$\hat{\mathcal{H}}(\hat{\mathbf{S}}) - \mathcal{H}(\hat{\mathbf{S}}) = \sum_{k=0}^{K} (\hat{h}_k - h_k)\hat{\mathbf{S}}^k = \sum_{k=0}^{K} (\hat{h}_k - h_k)(\mathbf{S}+\mathbf{E})^k \tag{46}$$

We are interested in characterizing how different are the outputs of $\hat{\mathcal{H}}(\hat{\mathbf{S}})$ and $\mathcal{H}(\hat{\mathbf{S}})$. Let $\sigma_{\max}(\mathbf{A})$ be the largest singular value of matrix $\boldsymbol{A}$; we know that $\|\boldsymbol{A}\boldsymbol{x}\| \leq \sigma_{\max}(\boldsymbol{A})\|\boldsymbol{x}\|$. Hence, for a given input, the norm of the difference of the output of the first layer is scaled at most by:

$$\begin{aligned}\sigma_{\max}(\hat{\mathcal{H}}(\hat{\mathbf{S}}) - \mathcal{H}(\hat{\mathbf{S}})) &= \max_i \left|\sigma_i\big(\hat{\mathcal{H}}(\hat{\mathbf{S}}) - \mathcal{H}(\hat{\mathbf{S}})\big)\right| \\ &= \max_i \left|\sum_{k=0}^{K} (\hat{h}_k - h_k)\sigma_i^k(\mathbf{S}+\mathbf{E})\right|.\end{aligned} \tag{47}$$

Furthermore, the following inequalities hold:

$$\begin{aligned}&\max_{\|\mathbf{E}\|\leq\epsilon} \sigma(\hat{\mathcal{H}}(\hat{\mathbf{S}}) - \mathcal{H}(\hat{\mathbf{S}})) = \max_{z\in\Sigma(\hat{\mathbf{S}})} \left|\sum_{k=0}^{K} (\hat{h}_k - h_k) z^k\right| \leq \\ &\max_{z\in\Sigma(\hat{\mathbf{S}})} \sum_{k=0}^{K} |\hat{h}_k - h_k||z|^k = \max_{0\leq x\leq\sigma_{\max}(\hat{\mathbf{S}})} \sum_{k=0}^{K} |\hat{h}_k - h_k| x^k.\end{aligned} \tag{48}$$

where $\Sigma(\hat{\mathbf{S}})$ denotes the set of singular values of $\hat{\mathbf{S}}$. Note that we can write $\hat{\mathbf{S}} = \mathbf{S} + \mathbf{E}$, and then Let $\boldsymbol{v}_{\max}$ be the largest right singular vector of $\hat{\mathbf{S}}$; we can write the inequalities:

$$\begin{aligned}\sigma_{\max}(\hat{\mathbf{S}}) = \max_{\boldsymbol{x}} \frac{\left\|\hat{\mathbf{S}}\boldsymbol{x}\right\|}{\|\boldsymbol{x}\|} &= \|(\mathbf{S}+\mathbf{E})\boldsymbol{v}_{\max}\| \\ &\leq \|\mathbf{S}\boldsymbol{v}_{\max}\| + \|\mathbf{E}\boldsymbol{v}_{\max}\| \leq \sigma_{\max}(\mathbf{S}) + \epsilon,\end{aligned} \tag{49}$$

Therefore, (48) can be further bounded as:

$$\max_{0\leq x\leq\sigma_{\max}(\hat{\mathbf{S}})} \sum_{k=0}^{K} |\hat{h}_k - h_k| x^k \leq \max_{0\leq x\leq\sigma_{\max}(\mathbf{S})+\epsilon} \sum_{k=0}^{K} |\hat{h}_k - h_k| x^k \tag{50}$$

This inequality holds due to the fact that the feasible set is relaxed. Moreover, the polynomial $\sum_{k=0}^{K} |\hat{h}_k - h_k| x^k$ has all positive coefficients, and thus is bounded by

$$\begin{aligned}&\max_{0\leq x\leq\sigma_{\max}(\mathbf{S})+\epsilon} \sum_{k=0}^{K} |\hat{h}_k - h_k| x^k \\ &\leq \sum_{k=0}^{K} |\hat{h}_k - h_k| (\sigma_{\max}(\mathbf{S}) + \epsilon)^k \\ &\leq \max\Big(1, (\sigma_{\max}(\mathbf{S}) + \epsilon)^K\Big) \sum_{k=0}^{K} |\hat{h}_k - h_k| \\ &= \overbrace{\max\Big(1, (\sigma_{\max}(\mathbf{S}) + \epsilon)^K\Big)}^{\gamma_1} \|\hat{\boldsymbol{h}} - \boldsymbol{h}\|_1\end{aligned} \tag{51}$$

Therefore, we can conclude that

$$\sigma_{\max}(\hat{\mathcal{H}}(\hat{\mathbf{S}}) - \mathcal{H}(\hat{\mathbf{S}})) \leq \gamma_1 \|\hat{\boldsymbol{h}} - \boldsymbol{h}\|_1 \tag{52}$$

This completes the proof. ■

B. Proof of Theorem 2

proof: To bound of $\rho(\mathcal{H}(\hat{\mathbf{S}}) - \mathcal{H}(\mathbf{S}))$, let $\mathbf{E} \triangleq \epsilon\bar{\mathbf{E}}$ and study the following expansion:

$$\begin{aligned}\mathcal{H}(\hat{\mathbf{S}}) - \mathcal{H}(\mathbf{S}) &= \sum_{k=0}^{K} h_k[(\mathbf{S}+\mathbf{E})^k - \mathbf{S}^k] \\ &= \sum_{k=1}^{K} h_k \sum_{\ell=1}^{k} \binom{k}{\ell} \mathbf{E}^\ell \mathbf{S}^{k-\ell} = \sum_{k=1}^{K} h_k \sum_{\ell=1}^{k} \binom{k}{\ell} \bar{\mathbf{E}}^\ell \mathbf{S}^{k-\ell} \epsilon^\ell,\end{aligned} \tag{53}$$

where it is easily to verify $\left\|\bar{\mathbf{E}}\right\| \leq 1$ due to $\|\mathbf{E}\| \leq \epsilon$. We can rewrite (53) as the following matrix polynomial function:

$$P(\epsilon) = \boldsymbol{A}_1\epsilon + \cdots + \boldsymbol{A}_K\epsilon^K \tag{54}$$

where the coefficients $\boldsymbol{A}_\ell, \forall \ell = 1, \cdots, K$ are expressed as follows:

$$\boldsymbol{A}_\ell = \sum_{k=\ell}^{K} h_k \binom{k}{\ell} \bar{\mathbf{E}}^\ell \mathbf{S}^{k-\ell}, \tag{55}$$

and in particular, we have $A_K = h_K \bar{\mathbf{E}}^K$. Therefore, we have the bound for the norm of $\boldsymbol{A}_\ell$:

$$\begin{aligned}\|\boldsymbol{A}_\ell\|_2 &\le \sum_{k=\ell}^{K}|h_k|\binom{k}{\ell}\left\|\mathbf{S}^{k-\ell}\right\|_2 \le \sum_{k=\ell}^{K}|h_k|\binom{k}{\ell}\|\mathbf{S}\|_2^{k-\ell}\\ &= \sum_{k=\ell}^{K}|h_k|\binom{k}{\ell}\sigma_{\max}^{k-\ell}(\mathbf{S}) \le \|\boldsymbol{A}\|_1\\ &\le \max_{1\le k\le K}|h_k|\sum_{k=1}^{K}\binom{k}{\ell}\sigma_{\max}^{k-\ell}\\ &\le \max_{1\le k\le K}|h_k|(1+\sigma_{\max}(\mathbf{S}))^K\end{aligned} \tag{56}$$

Consider the definition (19) $\gamma_2 \triangleq \max_{1\le k\le K}|h_k|(1+\sigma_{\max}(\mathbf{S}))^K$. We have that:

$$\|P(\epsilon)\|_2 \le \|\boldsymbol{A}_1\|\,\epsilon + \cdots + \|\boldsymbol{A}_K\|\,\epsilon^K \le \gamma_2\frac{\epsilon(1-\epsilon^K)}{1-\epsilon} \tag{57}$$

This completes the proof. ■

Lemma 2 *Considering the nonlinear activation function* $\mathrm{CReLU}(\cdot)$*, the distance between* $\mathrm{CReLU}(\mathcal{H}(\mathbf{S})\boldsymbol{x})$ *and* $\mathrm{CReLU}(\hat{\mathcal{H}}(\hat{\mathbf{S}})\boldsymbol{x})$ *is also bounded by:*

$$\begin{aligned}&\|\big(\,\mathrm{CReLU}(\hat{\mathcal{H}}(\hat{\mathbf{S}})\boldsymbol{x}) - \mathrm{CReLU}(\mathcal{H}(\mathbf{S})\boldsymbol{x})\big)\| \le\\ &\left(\gamma_1\left\|\hat{\boldsymbol{h}}-\boldsymbol{h}\right\|_1 + \gamma_2\frac{\epsilon(1-\epsilon^K)}{1-\epsilon}\right)\|\boldsymbol{x}\|\end{aligned} \tag{58}$$

Proof: Consider two complex numbers $z_1 = x_1 + \mathrm{j}y_1$ and $z_2 = x_2 + \mathrm{j}y_2$, the following relationship holds:

$$\begin{aligned}&|\mathrm{CReLU}(z_1) - \mathrm{CReLU}(z_2)| =\\ &|\mathrm{CReLU}(x_1+\mathrm{j}y_1) - \mathrm{CReLU}(x_2+\mathrm{j}y_2)| =\\ &|\mathrm{ReLU}(x_1-x_2) + \mathrm{j}\,\mathrm{ReLU}(y_1-y_2)| =\\ &\sqrt{(\mathrm{ReLU}(x_1-x_2))^2 + (\mathrm{ReLU}(y_1-y_2))^2} \le\\ &\sqrt{(x_1-x_2)^2+(y_1-y_2)^2} = |z_1-z_2|.\end{aligned} \tag{59}$$

The inequality follows from $|\mathrm{ReLU}(x_1-x_2)| \le |x_1-x_2|$. Then, we consider the distance between $\mathrm{CReLU}(\mathcal{H}(\mathbf{S})\boldsymbol{x})$ and $\mathrm{CReLU}(\hat{\mathcal{H}}(\hat{\mathbf{S}})\boldsymbol{x})$:

$$\begin{aligned}&\left\|\mathrm{CReLU}(\mathcal{H}(\mathbf{S})\boldsymbol{x}_t) - \mathrm{CReLU}(\hat{\mathcal{H}}(\hat{\mathbf{S}})\boldsymbol{x})\right\| \le\\ &\left\|\mathcal{H}(\mathbf{S})\boldsymbol{x} - \hat{\mathcal{H}}(\hat{\mathbf{S}})\boldsymbol{x}\right\|\\ &= \left\|\mathcal{H}(\mathbf{S}) - \hat{\mathcal{H}}(\hat{\mathbf{S}})\right\|\|\boldsymbol{x}\| \le \left\|\mathcal{H}(\mathbf{S}) - \hat{\mathcal{H}}(\hat{\mathbf{S}})\right\|\\ &\le \left(\gamma_1\left\|\hat{\boldsymbol{h}}-\boldsymbol{h}\right\|_1 + \gamma_2\frac{\epsilon(1-\epsilon^K)}{1-\epsilon}\right)\|\boldsymbol{x}\|.\end{aligned} \tag{60}$$

This completes the proof. ■

Likewise, we can easily verify that

$$\begin{aligned}&|\tanh(z_1) - \tanh(z_2)| =\\ &|\tanh(x_1+\mathrm{j}y_1) - \tanh(x_2+\mathrm{j}y_2)| =\\ &|\tanh(x_1-x_2) + \mathrm{j}\tanh(y_1-y_2)| =\\ &\sqrt{(\tanh(x_1-x_2))^2 + (\tanh(y_1-y_2))^2} \le\\ &\sqrt{|x_1-x_2|^2+|y_1-y_2|^2} = |z_1-z_2|.\end{aligned} \tag{61}$$

The last equation holds due to the 1-Lipschitz property of $\tanh(\cdot)$.

C. Proof of Lemma 1

Proof: We could express the multilayer neural networks, i.e., $\boldsymbol{y} = \Phi(\boldsymbol{x}, \mathbf{S}, \theta)$, as a function composition:

$$\begin{aligned}&g(\boldsymbol{x}) = \mathcal{H}(\mathbf{S})\boldsymbol{x} = \sum_{k=0}^{K} h_k\mathbf{S}^k\boldsymbol{x}, \quad \boldsymbol{x}^1 = \mathrm{CReLU}(g(\boldsymbol{x})),\\ &f(\boldsymbol{x}^1) = \Theta^{cplx}\boldsymbol{x}^1, \quad \boldsymbol{y} = \tanh(f(\boldsymbol{x}^1)).\end{aligned} \tag{62}$$

Therefore, we could have the following inequality:

$$\begin{aligned}\|\boldsymbol{y}-\hat{\boldsymbol{y}}\| &= \left\|\tanh(f(\boldsymbol{x}^1)) - \tanh(\hat{f}(\boldsymbol{x}^1))\right\|\\ &= \left\|\Theta^{cplx}\boldsymbol{x}^1 - \hat{\Theta}^{cplx}\hat{\boldsymbol{x}}^1\right\|\\ &= \left\|\Theta^{cplx}\boldsymbol{x}^1 - \Theta^{cplx}\hat{\boldsymbol{x}}^1 + \Theta^{cplx}\hat{\boldsymbol{x}}^1 - \hat{\Theta}^{cplx}\hat{\boldsymbol{x}}^1\right\|\\ &\le \left\|\Theta^{cplx}\boldsymbol{x}^1 - \Theta^{cplx}\hat{\boldsymbol{x}}^1\right\| + \left\|\Theta^{cplx}\hat{\boldsymbol{x}}^1 - \hat{\Theta}^{cplx}\hat{\boldsymbol{x}}^1\right\|.\end{aligned}$$

For the first part, we have

$$\begin{aligned}&\left\|\Theta^{cplx}\boldsymbol{x}^1 - \Theta^{cplx}\hat{\boldsymbol{x}}^1\right\| \le \left\|\Theta^{cplx}\right\|\\ &\left\|\mathrm{CReLU}(\mathcal{H}(\mathbf{S})\boldsymbol{x}) - \mathrm{CReLU}(\hat{\mathcal{H}}(\hat{\mathbf{S}})\boldsymbol{x})\right\| \le \sigma_{\max}(\Theta^{cplx})\\ &\left[\gamma_1\left\|\hat{\boldsymbol{h}}-\boldsymbol{h}\right\|_1 + \gamma_2\frac{\epsilon(1-\epsilon^K)}{1-\epsilon}\right]\|\boldsymbol{x}\|.\end{aligned} \tag{63}$$

The last inequality is due to Lemma 2. For the second part, we have

$$\begin{aligned}&\left\|\Theta^{cplx}\hat{\boldsymbol{x}}^1 - \hat{\Theta}^{cplx}\hat{\boldsymbol{x}}^1\right\| \le \left\|\Theta^{cplx} - \hat{\Theta}^{cplx}\right\|\\ &\left\|\mathrm{CReLU}(\hat{\mathcal{H}}(\hat{\mathbf{S}})\boldsymbol{x})\right\| \le \delta_{\mathbf{w}}\left\|\sum_{k=0}^{K}\hat{h}_k\hat{\mathbf{S}}^k\right\|\|\boldsymbol{x}\|\\ &= \delta_{\mathbf{w}}\left\|\sum_{k=0}^{K}\hat{h}_k(\mathbf{S}+\mathbf{E})^k\right\|\|\boldsymbol{x}\|.\end{aligned} \tag{64}$$

Similar to Theorem 1, we could bound $\left\|\sum_{k=0}^{K}\hat{h}_k(\mathbf{S}+\mathbf{E})^k\right\|$ as follows:

$$\begin{aligned}&\max_{\|\mathbf{E}\|\le\epsilon}\sigma(\sum_{k=0}^{K}\hat{h}_k(\mathbf{S}+\mathbf{E})^k) = \max_{z\in\Sigma(\hat{\mathbf{S}})}\left|\sum_{k=0}^{K}\hat{h}_k z^k\right| \le\\ &\max_{z\in\Sigma(\hat{\mathbf{S}})}\sum_{k=0}^{K}|\hat{h}_k||z|^k = \max_{0\le x\le\sigma_{\max}(\hat{\mathbf{S}})}\sum_{k=0}^{K}|\hat{h}_k|x^k \le \gamma_1\left\|\hat{\boldsymbol{h}}\right\|_1\end{aligned} \tag{65}$$

where $\Sigma(\hat{\mathbf{S}})$ denotes the set of singular values of $\hat{\mathbf{S}}$. Therefore, we have the bound of (64) as follows:

$$\delta_{\mathbf{w}}\left\|\sum_{k=0}^{K}\hat{h}_k(\mathbf{S}+\mathbf{E})^k\right\| \le \delta_{\mathbf{w}}\gamma_1\left\|\hat{\boldsymbol{h}}\right\|_1. \tag{66}$$

By adding (63) and (66), we have the bound for $\|\boldsymbol{y}-\hat{\boldsymbol{y}}\|$:

$$\begin{aligned}&\|\boldsymbol{y}-\hat{\boldsymbol{y}}\| \le \delta_{\mathbf{w}}\gamma_1\left\|\hat{\boldsymbol{h}}\right\|_1\|\boldsymbol{x}\| + \sigma_{\max}(\Theta^{cplx})\\ &\left[\gamma_1\left\|\hat{\boldsymbol{h}}-\boldsymbol{h}\right\|_1 + \gamma_2\frac{\epsilon(1-\epsilon^K)}{1-\epsilon}\right]\|\boldsymbol{x}\|\end{aligned} \tag{67}$$

This completes the proof. ■

REFERENCES

[1] H. Leung and S. Haykin, "The complex backpropagation algorithm," *IEEE Transactions on signal processing*, vol. 39, no. 9, pp. 2101–2104, 1991.

[2] C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal, "Deep complex networks," in *International Conference on Learning Representations*, 2018.

[3] J. Bassey, L. Qian, and X. Li, "A survey of complex-valued neural networks," *arXiv preprint arXiv:2101.12249*, 2021.

[4] R. Ramakrishna and A. Scaglione, "Grid-Graph Signal Processing (Grid-GSP): A Graph Signal Processing Framework for the Power Grid," *IEEE Trans. Signal Process.*, 2021.

[5] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs," *IEEE transactions on signal processing*, vol. 61, no. 7, pp. 1644–1656, 2013.

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

[7] X. Dong, D. Thanou, L. Toni, M. Bronstein, and P. Frossard, "Graph signal processing for machine learning: A review and new perspectives," *IEEE Signal processing magazine*, vol. 37, no. 6, pp. 117–127, 2020.

[8] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[9] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[10] R. Ramakrishna, H.-T. Wai, and A. Scaglione, "A user guide to low-pass graph signal processing and its applications: Tools and applications," *IEEE Signal Processing Magazine*, vol. 37, no. 6, pp. 74–85, 2020.

[11] T. Adali, P. J. Schreier, and L. L. Scharf, "Complex-valued signal processing: The proper way to deal with impropriety," *IEEE Transactions on Signal Processing*, vol. 59, no. 11, pp. 5101–5125, 2011.

[12] I. Jabłoński, "Graph signal processing in applications to sensor networks, smart grids, and smart cities," *IEEE Sensors Journal*, vol. 17, no. 23, pp. 7659–7666, 2017.

[13] A. Sperduti and A. Starita, "Supervised neural networks for the classification of structures," *IEEE Transactions on Neural Networks*, vol. 8, no. 3, pp. 714–735, 1997.

[14] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and deep locally connected networks on graphs," in *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

[15] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[16] 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*, 2018, pp. 3634–3640.

[17] 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, 2018, pp. 362–373.

[18] F. Gama, J. Bruna, and A. Ribeiro, "Stability properties of graph neural networks," *IEEE Trans. Signal Process.*, vol. 68, pp. 5680–5695, 2020.

[19] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[20] W. Liao, B. Bak-Jensen, J. R. Pillai, Y. Wang, and Y. Wang, "A review of graph neural networks and their applications in power systems," *Journal of Modern Power Systems and Clean Energy*, 2021.

[21] K. Chen, J. Hu, Y. Zhang, Z. Yu, and J. He, "Fault location in power distribution systems via deep graph convolutional networks," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 1, pp. 119–131, 2019.

[22] A. S. Zamzam and N. D. Sidiropoulos, "Physics-aware neural networks for distribution system state estimation," *IEEE Transactions on Power Systems*, vol. 35, no. 6, pp. 4347–4356, 2020.

[23] L. Cai, Z. Chen, C. Luo, J. Gui, J. Ni, D. Li, and H. Chen, "Structural temporal graph neural networks for anomaly detection in dynamic graphs," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 3747–3756.

[24] X. Ma, J. Wu, S. Xue, J. Yang, C. Zhou, Q. Z. Sheng, H. Xiong, and L. Akoglu, "A comprehensive survey on graph anomaly detection with deep learning," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[25] M. Liang, Y. Meng, J. Wang, D. L. Lubkeman, and N. Lu, "Feedergan: Synthetic feeder generation via deep graph adversarial nets," *IEEE Transactions on Smart Grid*, vol. 12, no. 2, pp. 1163–1173, 2020.

[26] M. B. Do Coutto Filho and J. C. S. de Souza, "Forecasting-aided state estimation—part i: Panorama," *IEEE Transactions on Power Systems*, vol. 24, no. 4, pp. 1667–1677, 2009.

[27] M. B. Do Coutto Filho, J. C. S. de Souza, and R. S. Freund, "Forecasting-aided state estimation—part ii: Implementation," *IEEE Transactions on Power Systems*, vol. 24, no. 4, pp. 1678–1685, 2009.

[28] L. Zhang, G. Wang, and G. B. Giannakis, "Power system state forecasting via deep recurrent neural networks," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 8092–8096.

[29] M. J. Hossain and M. Rahnamay-Naeini, "State estimation in smart grids using temporal graph convolution networks," in *2021 North American Power Symposium (NAPS)*. IEEE, 2021, pp. 01–05.

[30] O. Boyaci, M. R. Narimani, K. R. Davis, M. Ismail, T. J. Overbye, and E. Serpedin, "Joint detection and localization of stealth false data injection attacks in smart grids using graph neural networks," *IEEE Transactions on Smart Grid*, vol. 13, no. 1, pp. 807–819, 2021.

[31] S. Wang, S. Bi, and Y.-J. A. Zhang, "Locational detection of the false data injection attack in a smart grid: A multilabel classification approach," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8218–8227, 2020.

[32] Y. Wang, Z. Zhang, J. Ma, and Q. Jin, "Kfrnn: an effective false data injection attack detection in smart grid based on kalman filter and recurrent neural network," *IEEE Internet of Things Journal*, vol. 9, no. 9, pp. 6893–6904, 2021.

[33] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge university press, 2012.

[34] R. Ramakrishna and A. Scaglione, "On modeling voltage phasor measurements as graph signals," in *2019 IEEE Data Science Workshop (DSW)*. IEEE, 2019, pp. 275–279.

[35] A. Anis, A. Gadde, and A. Ortega, "Efficient sampling set selection for bandlimited graph signals using graph spectral proxies," *IEEE Transactions on Signal Processing*, vol. 64, no. 14, pp. 3775–3789, 2016.

[36] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, "Matpower: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Transactions on power systems*, vol. 26, no. 1, pp. 12–19, 2010.

[37] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.