# SPP-CNN: An Efficient Framework for Network Robustness Prediction

Chengpei Wu, Yang Lou, Lin Wang, Junli Li, Xiang Li, and Guanrong Chen

Abstract—This paper addresses the robustness of a network to sustain its connectivity and controllability against malicious attacks. This kind of network robustness is typically measured by the time-consuming attack simulation, which returns a sequence of values that record the remaining connectivity and controllability after a sequence of node- or edge-removal attacks. For improvement, this paper develops an efficient framework for network robustness prediction, the spatial pyramid pooling convolutional neural network (SPP-CNN). The new framework installs a spatial pyramid pooling layer between the convolutional and fully-connected layers, overcoming the common mismatch issue in the CNN-based prediction approaches and extending its generalizability. Extensive experiments are carried out by comparing SPP-CNN with three state-of-the-art robustness predictors, namely a CNN-based and two graph neural networksbased frameworks. Synthetic and real-world networks, both directed and undirected, are investigated. Experimental results demonstrate that the proposed SPP-CNN achieves better prediction performances and better generalizability to unknown datasets, with significantly lower time-consumption, than its counterparts.

*Index Terms*—Complex network, robustness, convolutional neural network, spatial pyramid pooling, prediction.

### I. INTRODUCTION

THE study of complex networks enhances our understanding of various real-world systems, ranging from natural to engineering, to biological, and to social networks [1]– [3]. Scientific investigations intend to reveal the essence and characteristics of networks, while engineering studies focus on their functions and applications, regarding such as connectivity [4]–[6], controllability [7]–[14], data transmission and communication abilities [15], [16], and so on. Today, random failures and malicious attacks take place in many engineering

Chengpei Wu and Junli Li are with the College of Computer Science, Sichuan Normal University, Chengdu 610066, China, and also with the Key Laboratory of System Control and Information Processing, Ministry of Education, Shanghai 200240, China (e-mail: chengpei.wu@hotmail.com, lijunli@sicnu.edu.cn).

Yang Lou is with the Department of Computer Science, National Yang Ming Chiao Tung University, Hsinchu 300, Taiwan (e-mail: felix.lou@ieee.org).

Lin Wang is with the Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China, and also with the Key Laboratory of System Control and Information Processing, Ministry of Education, Shanghai 200240, China (e-mail: wanglin@sjtu.edu.cn).

Xiang Li is with the Institute of Complex Networks and Intelligent Systems, Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University, Shanghai 201210, and also with the Department of Control Science and Engineering, Tongji University, Shanghai 200240, China (e-mail: lix2021@tongji.edu.cn).

Guanrong Chen is with the Department of Electrical Engineering, City University of Hong Kong, Hong Kong, China (e-mail: eegchen@cityu.edu.hk).

This manuscript has been submitted for potential publication. (Chengpei Wu and Yang Lou contributed equally to this work.)

(Corresponding author: Yang Lou)

and technological applications, which significantly degrade or even destroy the network normal functions. Therefore, it has become important and necessary to strengthen such network functions against attacks and failures [4]–[6], [17]–[23]. This leads to the concern of *network robustness*, which has different meanings in different scenarios, and here it refers to the ability of a network to sustain its normal functions when a fraction of the network nodes and/or edges failure due to attacks. In this paper, network robustness is studied with respect to both connectivity and controllability against destructive attacks and failures in the forms of node-removals.

The enhancement of network robustness depends on some reliable and efficient measures of the defined robustness [5], [6]. In general, network robustness can be measured in both *a priori* and *a posteriori* manners. *A priori* measures are referred to specific network features that can be evaluated and calculated without performing attack simulations. Widely-used *a priori* robustness measures include topological measures, for example betweenness centrality [24] and clustering coefficient [25], and spectral measures include such as natural connectivity [26], algebraic connectivity [27], effective resistance [28], and so on. Network spectra can be calculated using the network adjacency matrix and Laplacian matrix [29]. Although *a priori* measures are easy-to-access with lower computational cost and complexity [6], [22], they have limited scopes of applications [30].

In contrast, a posteriori robustness measures have intuitively clear meanings with a wider range of applicability, namely, a posteriori measures are applicable to any type of networks under any kind of attacks, and they return distinguishable measure values with respect to different attack strategies. Therefore, the practical *a posteriori* measures remain as the main approach in robustness studies for many real-world applications. A *posteriori* measures are quantified by recording a sequence of values that represent the remaining functionality (here, connectivity and controllability) of the network after a series of node- or edge-removal attacks. Evaluation of a posteriori measures is generally time-consuming, however, not only due to the iterative node- or edge-removal processes, but also because of the recalculation of the concerned network functions, such as the size of largest connected component [4], the number of needed driver nodes [7], [8], and the number of communicable node pairs [15], [16].

In practice, it is not always necessary to calculate the exact values of the network robustness as done in attack simulation. For example, during an optimization process, network robustness is reevaluated after each structural disturbance, e.g., edge rewiring [31]–[33]. It is very time-consuming to

perform an attack simulation each time, especially for largescale networks. Therefore, robustness prediction approaches are more desirable in such cases, which significantly reduce the cost and time complexity [34].

Network robustness prediction can be achieved by using either analytical or computational methods. In so doing, the time complexity is either constant [35] or increasing but significantly slower than that of the attack simulation [36]. Analytical approximations have a much narrower scope of applications to, e.g., the controllability robustness under random edge-attacks [35], [37], [38]. In contrast, machine learning algorithms, such as neural networks and random forest schemes, have no such limitation [39]. By combining several machine learning algorithms together, a surrogate ensemble can be formed for robustness prediction through an optimization process [34].

Regarding neural networks-based frameworks, deep neural networks are more powerful than canonical machine learning algorithms for efficiently processing network data. Successful application examples include critical node identification using deep reinforcement learning [40] and graph attention networks [41]. The convolutional neural network (CNN) [42]based prediction processes network data as gray-scale images [36], thereby fast approximating the robustness performances against different attacks. Prior knowledge is useful to further improve the prediction performances [43]. This straightforward approach requires downsampling and upsampling for the input data that are smaller and larger than the fixed input size of CNN, however, thus information distortion may be severe if the input network size is very large or very small. Graph neural networks (GNN) [44], [45] are able to compress and unify higher-dimensional network raw data to lower-dimensional representations. Therefore, GNN-based approaches are more tolerable to the input data-size changes, and so have better prediction performances [46]. Taking the connectivity robustness prediction as an example, the CNNbased approach needs an average run time that is 3.48% of the attack simulation, while the GNN-based approach requires up to 82.8% of the attack simulation [46].

In this paper, to overcome the aforemention mismatch issue between the various input sizes and the fixed input size in the CNN-based processing, and also to achieve a balance between the flexibility of processing different input sizes and the approximation speed in computation, a spatial pyramid pooling (SPP) [47] layer is installed into the network robustness predictor, which is demonstrated to be superior to the other CNN-based and GNN-based frameworks.

Specifically, the work and contributions of this paper are summarized as follows:

- SPP-CNN is proposed, which has a wider tolerance to different input-data sizes than the CNN- and GNN-based approaches [46], while maintaining fast approximation speed like the CNN-based approaches [36].
- SPP-CNN shows stronger generalizability than the other approaches on predicting the network robustness for datasets with unseen topologies and sizes.
- SPP-CNN demonstrates better performances than the other approaches on predicting the robustness of real-

2

world networks, with consistent advantages for both synthetic and real-world networks.

The rest of the paper is organized as follows. Section II reviews several measures of the network robustness against destructive node-removal attacks, and both CNN-based and GNN-based prediction frameworks. Section III introduces the proposed SPP-CNN. Section IV presents extensive empirical experiment results with comparison and analysis. Section V concludes the investigation.

#### **II. PRELIMINARIES**

In this paper, network robustness is considered from two specific aspects, namely the *a priori* connectivity robustness and controllability robustness, while other *a posteriori* robustness measures such as communication robustness can be investigated in a similar manner. Only node-removal attacks is discussed, while edge-attacks can also be studied in the same way. Three state-of-the-art network robustness predictors are reviewed, namely the CNN-based robustness predictor (CNN-RP) [36], PATCHY-SAN [48], and the learning feature representation-based predictor (LFR-CNN) [46]. Both PATCHY-SAN and LFR-CNN consist of a GNN module for graph representation learning, followed by a CNN for robustness performance prediction (regression).

#### A. Robustness Measures

1) Connectivity Robustness: In this paper, connectivity robustness refers to the ability of a network to maintain its connectivity against destructive node-removal attacks, which is widely measured by counting the size of the largest connected component (LCC) in the network after an attack. If an undirected network is connected or a directed network is weakly connected, then the LCC is the network itself; otherwise, a connected component that includes the largest number of nodes is an LCC. Connectivity robustness is evaluated by the following index:

$$R_1 = \sum_{i=0}^{N-1} r_1(i) = \sum_{i=0}^{N-1} \frac{N_L(i)}{N-i},$$
(1)

where N is the number of nodes in the given network before being attacked; i is the total number of nodes that have been removed from the network;  $N_L(i)$  is the number of nodes in the remaining LCC; therefore,  $r_1(i)$  is the density of nodes remaining in the *i*th LCC. The denominator N - i ensures  $R_1 \in (0, 1]$ , but if the denominator is replaced by N then  $R_1 \in (0, 0.5]$  as used in [4].

The series of values  $r_1(i)$  can be plotted to show a *connectivity curve*, for which the scalar  $R_1$  reflects the overall connectivity robustness against node-removal attacks: a higher  $R_1$  value indicates a better connectivity robustness.

2) Controllability Robustness: Controllability robustness reflects the ability of a networked system to maintain or regain its controllability with the lowest control cost, e.g., the minimum number of driver nodes (DN) that are needed to add to the network after the attack. For a general linear time-invariant (LTI) networked system  $\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}$ , where  $\mathbf{x} \in \mathbb{R}^N$ 

represents the state vector;  $\mathbf{u} \in \mathbb{R}^{b}$  represents the control input;  $A \in \mathbb{R}^{N \times N}$  and  $B \in \mathbb{R}^{N \times b}$  are constant matrices. This LTI system is *state controllable* if and only if there exist a control input  $\mathbf{u}$  that can drive the state  $\mathbf{x}$  to move from any initial state to any target state in the state space in finite time. The state controllability can be determined by checking whether the controllability matrix  $[B \ AB \ A^2B \ \cdots A^{N-1}B]$  has a full row-rank [49]. The concept of *structural controllability* is a slight generalization of the state controllability, to deal with two parameterized matrices A and B, in which the parameters characterize the structure of the underlying system in the sense that if there are specific parameter values that can ensure the system to be state controllable then the system is said to be structurally controllable.

For a network with many LTI systems, any node system with control input is a ND. The minimum number of NDs needed to retain the (state or structural) controllability of the network can be determined by using either the minimum inputs theorem (MIT) [7] for directed networks or the exact controllability theorem (ECT) [8] for both directed and undirected networks, which are defined as follows:

$$N_D = \begin{cases} \max\{1, N - |E|\}, & \text{using MIT [7]}, \\ \max\{1, N - \operatorname{rank}(A)\}, & \text{using ECT [8]}, \end{cases}$$
(2)

where |E| represents the number of edges in the maximum matching E, which is a basic concept in graph theory [7]. Under a sequence of node-removal attacks, the controllability robustness is measured by

$$R_2 = \sum_{i=0}^{N-1} r_2(i) = \sum_{i=0}^{N-1} \frac{N_D(i)}{N-i},$$
(3)

where  $N_D(i)$  is the number of DNs and  $r_2(i)$  is the density of DNs, which are needed to retain the network controllability after a total of *i* nodes have been removed by the attack. Similarly, the series of values  $r_2(i)$  can be plotted to show a *controllability curve*, where  $R_2$  measures the overall controllability robustness: a lower  $R_2$  values indicates a better connectivity robustness.

#### **B.** Network Robustness Predictors

There are three commonly-used network robustness predictors, namely CNN-RP [36], PATCHY-SAN [48], and FR-CNN [46]. Basic principles as well as the pros and cons of these approaches are reviewed and discussed in this subsection.

The general structures of CNN-RP, PATCHY-SAN, and LFR-CNN are visualized in Fig. 1. The input is the adjacency matrix, which could also be a Laplacian matrix or other representations, and the output is the predicted robustness performance.

1) CNN-RP: The structure of CNN-RP [36], [50], using a VGG-structured CNN [51], is shown in Fig. 2. Adjacency matrices are treated as gray-scale images and processed by CNN directly. Classification and regression tasks are completed using such an image-processing mechanism [43].

The output of CNN-RP is an N-vector, which represents a connectivity or controllability curve, denoted by  $\hat{v}$ . The mean-



Fig. 1: General frameworks of CNN-RP, PATCHY-SAN, and LFR-CNN. CNN-RP processes network data as gray-scaled images, while for PATCHY-SAN and LFR-CNN, the LFR module performs the selection, assembly, and normalization (SAN) operations, which compress higher-dimensional (HD) network data to be lower-dimensional (LD) representations.



Fig. 2: CNN structure in CNN-RP. The input is an adjacency matrix; the output is an *N*-vector. Seven feature map (FM) groups are generated with  $N_i = \lceil N/2^{(i+1)} \rceil$ , for i = 1, 2, ..., 7. Concatenation layer reshapes the data to be a vector, from FM 7 to FC<sub>1</sub>. FC<sub>1</sub>=512 $N_7^2$  and FC<sub>2</sub>=4096 [36].

squared error between the predicted curve  $\hat{v}$  and the true curve v is used as the loss function:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} ||\hat{v}(i) - v(i)||, \qquad (4)$$

where  $\hat{v}(i)$  and v(i) represent the predicted and the true connectivity or controllability values, respectively; *i* is the total number of nodes that have been removed from the network;  $|| \cdot ||$  is the Euclidean norm. The true values v(i) are obtained by performing attack simulation. The training process aims at adjusting the internal parameters, aiming at minimizing  $\mathcal{L}$ .

2) PATCHY-SAN and LFR-CNN: As shown in Fig. 1, PATCHY-SAN and LFR-CNN share the same LFR module, which converts higher-dimensional network data to lowerdimensional representations. This conversion is achieved by using a series of three operations: selection, assembly, and normalization (SAN).

First, the N network nodes are sorted according to their importance, which can be quantified by certain node centrality measure such as node degree or node betweenness. A total of W most important nodes are first selected. For each selected node, a receptive field of size g is created based on its neighboring information. If N < W, then all-zero receptive fields are added for padding. Next, a breadth-first search is conducted to construct the neighborhood field for each selected node. Finally, a normalization step converts and normalizes the neighboring field for each selected node to an embedded vector with a uniform length gh, where h is the number of attributes used for the neighboring nodes. The first element in the resultant normalized vector represents the root node, followed by the neighboring nodes sorted according to their centrality measures.

To this end, an N-node network represented by an  $N^2$  adjacency matrix has been represented by a  $W \times (gh)$  matrix, where there are W receptive fields and each receptive field is presented by a  $1 \times gh$  vector. Since  $g \ll N$  and  $h \ll N$ , this procedure generates lower-dimensional learned feature representations from higher-dimensional raw network data. Then, the lower-dimensional representations are processed by CNN to predict the robustness performances of the given networks.

For PATCHY-SAN, a shallow 1D-CNN structure is employed, while for LFR-CNN a VGG-structured [51] 2D-CNN with 3 feature maps (FMs) is used. The numbers of internal parameters to be adjusted during training are  $5.1 \times 10^5$  and  $6.0 \times 10^6$  for PATCHY-SAN and LFR-CNN, respectively.

The same mean-squared error between the predicted curve  $\hat{v}$  and the true curve v as shown in Eq. (4) is used as the loss function for both PATCHY-SAN and LFR-CNN.

#### C. Error Measures

Let  $\mathbf{v_t} = \{v_t(i)\}_{i=0}^{N-1}$  and  $\mathbf{v_p} = \{v_p(i)\}_{i=0}^{N-1}$  be the true and the predicted robustness curves obtained by attack simulation, respectively. The *prediction error*  $\xi$  is calculated by

$$\xi = \frac{1}{N} \sum_{i=0}^{N-1} \xi(i) , \qquad (5)$$

where  $\xi(i) = |v_t(i) - v_p(i)|$ , i = 0, 1, ..., N - 1. Given two prediction errors obtained by two different robustness predictors, denoted by  $\xi_1$  and  $\xi_2$ , if  $\xi_1 < \xi_2$  then the first predictor performs better than the second.

#### III. SPATIAL PYRAMID POOLING

A CNN consists of convolutional layers and fully-connected layers, as shown in Fig. 2. The convolutional layers, which work in a sliding-window manner, are flexible with different input sizes, while the nature of the fully-connected layers requires a predefined fixed size of input. Therefore, CNNs require a fixed size for the input data, e.g.,  $224 \times 224$ . For image processing tasks such as classification and object detection [51]–[53], cropping [54] and warping [55], the images may be fit to the required fixed input size. When complex network data are used as the input to CNNs, upsampling or subsampling is suitable [36], since cropping and warping may discard or distort some specific marginal regions of an image, while upsampling and subsampling cause uniformly random information loss or distortion. However, resizing of network data not only will change the true network size, but also will change the true topology, by removing existing nodes or adding dummy nodes. But the resized network data can be misleading, for example, when hub nodes are deleted.

On the other hand, the GNN-based approaches can effectively reduce the information loss by extracting neighboring information from the important nodes, as discussed in Subsection II-B2. The representation learning process converts the input higher-dimensional data of any size to a regulated fixedsized lower-dimensional representation, and then passes it to a CNN. This process significantly widen the application range of the robustness predictor to different network sizes [46]. However, the GNN-based approaches are significantly slower than the CNN-based approaches, where the representation learning process is the most time-consuming part.

Embedding a spatial pyramid pooling (SPP) [47] layer into CNN, in between the convolutional layers and fully-connected layers, brings some benefits: it brings a buffer layer from the flexible convolutional layers to the fixed fully-connected layers. SPP is developed from the canonical spatial pyramid matching algorithm in computer vision [56], [57], which statistically counts the feature distributions of images from multiple scales, such that better recognition performances can be achieved. The spatial pyramid matching algorithm sets a number of spatial bins, and then local features within each individual spatial bin are captured statistically. The representation of the whole image consists of local features from different spatial bins.

As an extension of the spatial pyramid matching algorithm, SPP uses max pooling instead of statistical counting to local feature learning and resizing. Given input images of different sizes, the convolution layers are able to process all the information and then generate feature maps of different sizes. The SPP layer then transforms these feature maps to fixed-length representations.

As shown in Fig. 3, an  $N \times N$  input image results in L $N' \times N'$  feature maps, where L is the number of filters in the last convolutional layer. In the SPP layer, feature maps are divided into 3 different levels of spatial bins, of sizes  $1 \times 1$ ,  $2 \times 2$ , and  $4 \times 4$ , which are processed by max pooling with corresponding sizes. Then, a representation vector of size pLis generated as the output of the SPP layer. Here, both L and p are pre-defined hyperparameters. As a result, for the input image of any size, a fixed length pL-vector is generated as the input to the fully-connected layers. In this paper, three pyramid pooling levels are used, with sizes of  $1 \times 1$ ,  $2 \times 2$ and  $4 \times 4$ , respectively. It has been empirically verified that the performance of the SPP layer is insensitive to different settings of the pyramid bins [47].

By installing such an SPP layer in CNN, an SPP-CNN is constructed for network robustness prediction. Fig. 4 shows the structure of SPP-CNN, which consists of 6 convolutional layers, 1 SPP layer, and 3 fully connected layers. This structure is able to predict the robustness performances of networks with hundreds to thousands of nodes. Specifically, p = 21represents the total number of bins, and L = 256 denotes the number of filters in the last convolutional layer. The resultant representation has a fixed-length of  $21 \times 256 = 5376$ , which is also the fixed input size of the first fully-connected layer (FC<sub>1</sub>), namely  $L_1 = pL = 5376$ .

The parameter setting of the convolutional layers is shown in Table I. Source codes of this work are available for the public<sup>1</sup>.

<sup>1</sup>https://fylou.github.io/sourcecode.html



5



spatial pyramid pooling layer

Fig. 3: The convolutional neural network structure with a spatial pyramid pooling layer. The detailed structure of convolutional layers are shown in Fig. 4.



Fig. 4: CNN structure of SPP-CNN. The spatial pyramid pooling layer is installed between the convolutional layers and fully-connected layers. Hard-sigmoid is installed in the last fully-connected layer, while in other layers ReLU is installed.

TABLE I: Parameter setting of the convolutional layers in SPP-CNN.

Group	Layer	Kernel size	Stride	Output channel
Group 1	Conv7-64	$7 \times 7$	1	64
	Max2	$2 \times 2$	2	64
Group 2	Conv5-64	$5 \times 5$	1	64
	Max2	$2 \times 2$	2	64
Group 3	Conv3-128	$3 \times 3$	1	128
	Max2	$2 \times 2$	2	128
Group 4	Conv3-128	$3 \times 3$	1	128
	Max2	$2 \times 2$	2	128
Group 5	Conv3-256	$3 \times 3$	1	256
	Max2	$2 \times 2$	2	256
Group 6	Conv3-256	$3 \times 3$	1	256
	Max2	$2 \times 2$	2	256

#### **IV. EXPERIMENTAL STUDIES**

In this section, the proposed SPP-CNN is experimentally tested and compared with CNN-RP [36], PATCHY-SAN [48], and LFR-CNN [46]. Network connectivity robustness under maximum-degree node attacks and controllability robustness under random node attacks are predicted. Other network robustness under different attack strategies can be studied in the same manner. Both directed and undirected, and synthetic and real-world networks are simulated. All experiments are performed on a PC with GeForce RTX 3080 GPU, which has memory (RAM) 10 GB with running the Ubuntu 20.04.3 LTS

Operating System.

This section is organized as follows. Subsection IV-A introduces the experimental settings. General comparison of robustness prediction performances are presented in Subsection IV-B, which demonstrates that SPP-CNN performs as well as other state-of-the-art GNN-based robustness predictors when the test data drawn from the same distribution are used for training. The run-time comparison presented in Subsection IV-C demonstrates that SPP-CNN is significantly faster than the GNN-based predictors. Then, the generlizability of the predictors is verified in Subsections IV-D and IV-E, where the test datasets with unseen synthetic and real-world networks are tested. Finally, the pros and cons of different predictors are discussed in Subsection IV-F.

#### A. Experimental Settings

Nine representative synthetic network models are used for simulation, including Barabási–Albert (BA) scale-free [58], [59], extreme homogeneous (EH) [60], Erdös-Rényi (ER) random-graph [61], q-snapback (QS) [62], random hexagon (RH) [63], random triangle (RT) [63], generic scale-free (SF) [64], Newman–Watts small-world (SW-NW) [65], and Watts– Strogatz small-world (SW-WS) [25] network models.

Specifically, the generation of BA networks is based on preferential attachment [58]; SF networks are generated using predefined weights, namely the probability of connecting two nodes *i* and *j* is proportional to their weights  $w_i$  and  $w_j$ , where  $w_i = (i + \theta)^{-\sigma}$  for any node *i*, parameters  $\sigma \in [0, 1)$  and  $\theta \ll N$ ; EH networks are generated by performing random edge rectifications onto ER networks, such that the degree distribution of EH is extremely homogeneous [60]; RT and RH consist of random triangles and hexagons, respectively [63]. These models, BA, EH, ER, RH, RT, SF, generate undirected instances by default, while directed instances are generated by assigning random directions onto the edges.

QS consists of a directed backbone chain and multiple snapback edges [62]; SW starts from an N-node directed loop having K (K = 2 here) connected nearest-neighbors; shortcuts are randomly added without removing any existing edges in

TABLE II: Comparison of average prediction errors among SPP-CNN, CNN-RP, PATCHY-SAN, and LFR-CNN. Signs in parentheses denote the Kruskal-Wallis H-test [66] results. A '+' sign denotes that SPP-CNN is superior to the other methods with smaller errors; a ' $\approx$ ' sign denotes no significant difference; while a '-' sign denotes that SPP-CNN is inferior to the other methods with greater errors.

			BA	EH	ER	QS	RH	RT	SF	SW- NW	SW- WS
Connectivity Robustness Eq. (1) Controllability Robustness Eq. (3)	Connectivity	SPP-CNN	0.038	0.057	0.032	0.067	0.056	0.105	0.017	0.030	0.031
	Pobustness	CNN-RP	0.146(+)	0.138(+)	0.129(+)	0.162(+)	0.121(+)	0.118(≈)	0.068(+)	0.094(+)	0.099(+)
	$E_{a}$ (1)	PATCHY-SAN	0.040(+)	0.032(-)	0.024(-)	0.027(-)	0.028(-)	0.031(-)	0.029(+)	0.027(≈)	0.024(-)
	Eq. (1)	LFR-CNN	0.041(+)	0.078(+)	0.039(≈)	0.076(≈)	0.023(-)	0.031(-)	0.018(≈)	0.029(≈)	0.027(≈)
	Controllability	SPP-CNN	0.031	0.025	0.025	0.042	0.021	0.030	0.025	0.028	0.025
	Pobustness	CNN-RP	0.092(+)	0.040(+)	0.062(+)	0.104(+)	0.051(+)	0.060(+)	0.134(+)	0.054(+)	0.059(+)
	$E_{a}$ (3)	PATCHY-SAN	0.032(≈)	0.028(≈)	0.026(≈)	0.030(-)	0.019(≈)	0.024(≈)	0.049(+)	0.024(≈)	0.022(≈)
	Eq. (3)	LFR-CNN	0.024(≈)	0.013(-)	0.018(-)	0.015(-)	0.015(-)	0.019(-)	0.039(+)	0.015(-)	0.015(-)
	Connectivity	SPP-CNN	0.023	0.070	0.025	0.026	0.028	0.047	0.016	0.029	0.028
Connectivity Robustness Eq. (1) Controllability Robustness Eq. (3)	Pobustness	CNN-RP	0.055(+)	0.188(+)	0.121(+)	0.135(+)	0.121(+)	0.100(+)	0.020(≈)	0.149(+)	0.140(+)
	$E_{a}$ (1)	PATCHY-SAN	0.030(+)	0.032(-)	0.022(-)	0.024(≈)	0.022(-)	0.026(-)	0.025(+)	0.025(≈)	0.020(-)
	Eq.(1)	LFR-CNN	0.026(≈)	0.066(≈)	0.025(≈)	0.027(≈)	0.019(-)	0.022(-)	0.016(≈)	0.023(≈)	0.020(-)
	Controllability	SPP-CNN	0.027	0.013	0.016	0.014	0.018	0.018	0.026	0.016	0.017
	Pobustness	CNN-RP	0.060(+)	0.037(+)	0.031(+)	0.035(+)	0.039(+)	0.039(+)	0.085(+)	0.045(+)	0.046(+)
	Eq. (3)	PATCHY-SAN	0.028(≈)	0.015(≈)	0.020(≈)	0.017(+)	0.021(+)	0.020(≈)	0.040(+)	0.016(≈)	0.016(≈)
	LFR-CNN	0.037(+)	0.016(+)	0.017(≈)	0.017(+)	0.018(≈)	0.020(≈)	0.060(+)	0.018(+)	0.019(≈)	

SW-NW [65], while rewiring operations are performed in SW-WS [25]. Thus, QS, SW-NW, and SW-WS generate directed network instances by default, while undirected instances are generated by removing the edge directions.

The average degree of each network instance is assigned reasonably at random. For directed networks, the average degree range is set as  $\langle k \rangle \in [2.5, 5]$  for the two SW models,  $\langle k \rangle \in [2, 4]$  for RH,  $\langle k \rangle \in [1.5, 3]$  for RT, and for the other models,  $\langle k \rangle \in [3, 6]$ . For undirected networks, the average degree range is set to double, namely, the range is set as  $\langle k \rangle \in [5, 10]$  for the two SW models,  $\langle k \rangle \in [4, 8]$  for RH,  $\langle k \rangle \in [3, 6]$  for RT, while for the other models,  $\langle k \rangle \in [6, 12]$ .

Define three sets of synthetic network models,  $S_1 = \{BA, EH, ER, QS, RH, RT, SF, SW-NW, SW-WS\}$ ,  $S_2 = \{ER, QS, SF, SW-NW\}$ , and  $S_3 = \{BA, EH, RH, RT, SW-WS\}$ . Three network size ranges are set as  $N_a \in [700, 1300]$ ,  $N_b \in [300, 700]$ ,  $N_c \in [1300, 1700]$ . For each synthetic model, 1000 network instances are randomly generated as the training data, and 100 instances as the test data. Since the structural connectivity and controllability are independent of the edge weights, only unweighted networks are simulated here.

For the LFR module in both PATCHY-SAN and LFR-CNN, the number of features is set as h = 2, where node degree and clustering coefficient are employed; the size of receptive field is set as g = 10, both are the same as that in [46], [48].

#### B. Comparison of Prediction Performances

Both training and test data are drawn from the same sample space, namely each network instance is randomly generated from any of the 9 synthetic models in  $S_1$ , with a network size randomly picked as  $N_a \in [700, 1300]$ .

Table II shows the average prediction errors obtained by SPP-CNN, CNN-RP, PATCHY-SAN, and LFR-CNN. Boxplots of the prediction errors are presented in Figs. S1–S4 of the Supplementary Information (SI)<sup>2</sup> due to space limitation here. Network robustness in terms of connectivity and controllability

is predicted, and the prediction errors are calculated using Eq. (5). The errors are averaged from 100 independent runs for each synthetic model. A total of  $2 \times 2 \times 9 = 36$  comparisons are performed between SPP-CNN and each of CNN-RP, PATCHY-SAN, and LFR-CNN, namely, directed and undirected networks, two robustness measures, and 9 synthetic network models. The Kruskal-Wallis H-test [66] results are shown with the corresponding prediction error values, where a '+' sign denotes that SPP-CNN performs significantly better than the other methods with smaller errors; a ' $\approx$ ' sign denotes no significant difference between SPP-CNN and the other methods; while a '-' sign denotes that SPP-CNN performs significantly worse than the other methods with greater errors.

SPP-CNN performs significantly better than CNN-RP in 34 cases, and for the rest 2 cases, there is no significant difference between them. As for PATCHY-SAN and LFR-CNN, the same numbers of superiors and inferiors are obtained, showing that SPP-CNN significantly outperforms PATCHY-SAN (or LFR-CNN) in 8 cases; SPP-CNN performs significantly worse than PATCHY-SAN (or LFR-CNN) in 12 cases; and they perform statistically equally in the rest 16 cases.

In a nutshell, it is clear that SPP-CNN performs equivalently to or marginally worse than PATCHY-SAN and LFR-CNN, but significantly better than CNN-RP.

#### C. Run Time Comparison

The powerful GNN-based representation learning part of PATCHY-SAN and LFR-CNN significantly enhances the precision of robustness prediction, which however is the most time-consuming part. In contrast, SPP-CNN does not involve any GNN-based operation for feature learning.

Figure 5 shows the run time comparison of SPP-CNN, CNN-RP, PATCHY-SAN, LFR-CNN, and attack simulation (SIM). Fig. 5 (a) presents the run time when the network size is  $N_a \in [700, 1300]$ , of which the average network size is 1000; for Fig. 5 (b), the network size is set as  $N_b \in [700, 1300]$ , of which the average network size is 500.



Fig. 5: Run time comparison of SPP-CNN, CNN-RP, PATCHY-SAN, LFR-CNN, and attack simulation (SIM): (a) for network size  $N_a \in [700, 1300]$ ; and (b) for network size  $N_b \in [300, 700]$ 

For each subplot in Fig. 5, it is clear that CNN-RP is the fastest, followed by SPP-CNN. Clearly, PATCHY-SAN and LFR-CNN are significantly slower than CNN-RP and SPP-CNN, but much faster than attack simulations. For PATCHY-SAN and LFR-CNN, the representation learning takes most of the run time.

Comparing Figs. 5 (a) and (b), when the average network size is doubled from subplot (b) to subplot (a), the time consumption of SPP-CNN, CNN-RP, PATCHY-SAN, LFR-CNN, and SIM is increased by 1.61, 1,34, 4.78, 4.81, and 8.21 times, respectively. It is clear that the time consumption of CNN-RP and SPP-CNN increase significantly slower than the GNN-based predictors do.

The run time difference between CNN-RP and SPP-CNN is negligible, while SPP-CNN significantly outperforms CNN-RP, as can be seen from Table II, which also shows that the performance different between SPP-CNN and the GNN-based predictors is marginal, while the time complexity of SPP-CNN is significantly lower than PATCHY-SAN and LFR-CNN, as shown in Fig. 5.

#### D. Comparison of Generalizability

The generalizability of robustness predictors is tested from two aspects: 1) using the network models in  $S_1$ , two sets of test instances with unseen network sizes (UNS) are tested, namely, the network size of training data is drawn from  $N_a \in$ [700, 1300], while the test data are from  $N_b \in$  [300, 700] and  $N_c \in$  [1300, 1700]. 2) Given network size drawn from  $N_a \in$ [700, 1300], training instances are generated from the network models in  $S_2$ , while the test instances are from  $S_3$ . Since  $S_2$ and  $S_3$  are two mutually exclusive subsets of  $S_1$ ,  $S_3$  is called the test data of unseen network topology (UNT). Note that although  $S_2$  and  $S_3$  are mutually exclusive, there are some similarities between the models in the two sets, e.g., SW-NW  $\in S_2$  and SW-WS  $\in S_3$ .

Table III summarizes the comparison of the significant differences between SPP-CNN and each of CNN-RP, PATCHY-SAN, and LFR-CNN. The prediction errors for UNS and UNT are presented in Figs. S5–S8 and S13 of SI, respectively. The barcharts of the numbers of significant performance differences for UNS and UNT are shown in Figs. S9–S12 and

TABLE III: Comparison of significant difference between SPP-CNN and each of CNN-RP, PATCHY-SAN, and LFR-CNN.

	Significant Difference	(+)	(-)	(≈)
UNS	SPP-CNN vs CNN-RP	266	12	10
	SPP-CNN vs PATCHY-SAN	169	78	41
	SPP-CNN vs LFR-CNN	124	108	56
UNT	SPP-CNN vs CNN-RP	11	2	7
	SPP-CNN vs PATCHY-SAN	15	3	2
	SPP-CNN vs LFR-CNN	14	3	3

S14 of SI, respectively. For UNS, there are  $2 \times 2 \times 9 \times 8 = 288$ neck-to-neck comparisons between SPP-CNN and each of CNN-RP, PATCHY-SAN, and LFR-CNN, namely, directed and undirected, connectivity and controllability robustness, 9 synthetic models in  $S_1$ , and 8 UNS sections in  $N_b$  and  $N_c$ . For UNT, there are  $2 \times 2 \times 5 = 20$  comparisons, namely, directed and undirected, connectivity and controllability robustness, and 5 synthetic models in  $S_3$ .

Only when testing on UNS, SPP-CNN outperforms LFR-CNN marginally, but for all the rest comparisons, SPP-CNN gains more superiors than inferiors. This indicates the excellent generalizability of SPP-CNN.

#### E. Predicting Robustness for Real-world Networks

The prediction performance is studied by two experiments. The first experiment investigates the performances of SPP-CNN, PATCHY-SAN, and LFR-CNN on predicting the robustness of the tested real-world networks. The second experiment uses mixed sets of both synthetic and real-world networks as both training and test data. Define  $S_r = \{\text{Reddit-Multi-}12\text{K}^3 \text{ Networks}\}$ , where there are 1000 real-world networks selected randomly, of which the network size ranges as  $N_r \in [300, 700]$ .

1) Real-world Networks Only: Figure 6 (a) shows the prediction errors obtained by SPP-CNN, PATCHY-SAN, and LFR-CNN, where the training data set is  $S_r$  while the test set constitutes of other randomly picked 100 networks from Reddit-Multi-12K. The boxplot in Fig. 6 (a) shows that SPP-CNN outperforms both PATCHY-SAN and LFR-CNN, with statistic significance, using the Kruskal-Wallis test [66].

2) Real-world and Synthetic Networks: To clear up the doubt that the excellent performance of SPP-CNN on predicting real-world networks is random or due to overfitting,  $S_r$  and  $S_2$  are mixed together in experiments. Thus, the trained SPP-CNN is neither specialized for synthetic models nor for real-world networks.

As shown in Fig. 6 (b), the prediction error boxplot obtained by SPP-CNN on real-world networks is similar to that in Fig. 6 (a). This clears up the doubt of overfitting. Also, the prediction errors obtained by SPP-CNN for synthetic networks are similar to that in Fig. 6 (c), where a SPP-CNN is specifically trained for the 4 synthetic models.

In a nutshell, when real-world and synthetic networks are mixed and used as both training and test data, SPP-CNN can perform robustness prediction as excellent as the cases where



Fig. 6: Prediction error comparison in the form of boxplot: (a) both training and test data are real-world networks; (b) for SPP-CNN, the training and test data include both real-world networks  $S_r$  and synthetic models  $S_2$ ; (c) the benchmark performance of SPP-CNN when only  $S_2$  is used as both training and test data.

TABLE IV: A summary of the performances of SPP-CNN when different datasets (including  $S_1$ ,  $S_2$ ,  $S_3$ , and  $S_r$ ) and network sizes (including  $N_a$ ,  $N_b$ , and  $N_c$ ) are used. A ' $\approx$ ' sign represents that the numbers of superiors and inferiors obtained by SPP-CNN are similar to that obtained by the compared method; while a ' $\succ$ ' sign means that SPP-CNN obtains clearly more superiors than inferiors in the corresponding comparison.

Training data	$S_1(N_a)$	$S_1(N_a)$	$S_2(N_a)$	$S_r$	$S_r + S_2$
Test data	$S_1(N_a)$	$S_1 (N_b \text{ and } N_c)$	$S_3(N_a)$	$S_r$	$S_r + S_2$
Performance of SPP-CNN	≈ PATCHY-SAN ≈ LFR-CNN ≻CNN-RP	$\succ PATCHY-SAN \\ \approx LFR-CNN \\ \succ CNN-RP$	$\succ PATCHY-SAN$ $\succ LFR-CNN$ $\succ CNN-RP$	$\succ PATCHY-SAN$ $\succ LFR-CNN$ $\succ CNN-RP$	as good as when $S_r$ and $S_2$ are trained and tested separately

only real-world or only synthetic networks are used. This also implies the excellent generalizability of SPP-CNN.

#### F. Discussions



Fig. 7: Overall performances of SPP-CNN, CNN-RP, PATCHY-SAN, and LFR-CNN.

1) Overall Performance: Table IV shows the overall performances of SPP-CNN compared to CNN-RP, PATCHY-SAN, and LFR-CNN. The results are summarized from the results presented in Subsections IV-B, IV-D, and IV-E. When the test data and training data are drawn from the same sample space, SPP-CNN performs as good as the GNN-based predictors. However, if the test data are drawn from different sample spaces, SPP-CNN consistently outperforms the other predictors. Here, different sample spaces mean that either UNS or UNT is set. Also, SPP-CNN outperforms the GNN-based predictors on predicting the robustness of real-world networks.

Considering both generalizability and run time, Fig. 7 shows the performance comparison in two-dimensional coordinate plots. Clearly, SPP-CNN and CNN-RP are faster, while SPP-CNN meanwhile possesses the best generalizability.

2) Information Loss: Given an  $N \times N$  adjacency matrix as input, CNN-RP requires a fixed input size  $W \times W$ , and thus downsampling or upsampling is necessary. If N > W, then N - W columns and rows are needed to be randomly deleted from the adjacency matrix in order to fit the input size. If N < W, then W - N empty columns and rows are needed to be randomly added for padding. These deletion or addition operations may significantly distort the original network topology, and thus degenerate the subsequent robustness prediction performance.

For PATCHY-SAN and LFR-CNN, as discussed in Subsection II-B2, if N > W, then only the neighboring fields of the W most important nodes are selected to construct the receptive fields. While if N < W then W - N dummy nodes are generated. For CNN-RP, there is always a proportion of  $\delta = \frac{|N-W|}{N}$  information distortion, while for PATCHY-SAN and LFR-CNN, this information distortion is significantly lower than  $\sigma$ , since the neighboring fields of the W most important nodes are always included, but other unimportant nodes may also be compressed into the neighboring fields.

Finally, for SPP-CNN, there is always no information loss or distortion from the input, since networks of any sizes are able to be input without the need of resizing. Moreover, local features are better captured by the SPP layer.

#### V. CONCLUSIONS

Measuring network robustness by attack simulations is time-consuming, while deep neural networks provide a more cost-effective technique for robustness prediction, which can replace the iterative attack simulations, at least partially. The CNN-based framework CNN-RP can predict network robustness fast, but is inefficient when the size of the concerned network is different from the fixed input size of the CNN. On the other hand, PATCHY-SAN and LFR-CNN, which incorporate both GNN and CNN, are able to predict network robustness of different sizes and various topologies with low prediction errors. However, these GNN-based robustness predictors perform significantly slower than CNN-RP, due to the powerful but time-consuming feature learning module installed.

In this paper, to overcome the mismatch issue between the various network sizes and the somewhat fixed input size of the CNN, a spatial pyramid pooling layer is installed between the convolutional and fully-connected layers of the CNN, yielding the new SPP-CNN framework.

Extensive experiments are carried out by comparing SPP-CNN with CNN-RP, PATCHY-SAN, and LFR-CNN. Three sets of synthetic (directed and undirected) networks with three different network size ranges, together with one set of real-world networks, are simulated. Detailed comparisons are performed, where both training and test data are drawn from the same sample space. Generalization abilities of the predictors are also examined, where the test data are drawn from different sample spaces, other than the training data space. The prediction performances on real-world networks are tested from two aspects: 1) real-world networks are trained and tested separately, and 2) real-world networks are mixed with synthetic networks. All the experimental results demonstrate the excellent performances of SPP-CNN: 1) SPP-CNN achieves significantly better prediction performances than CNN-RP with similar performances as PATCHY-SAN and LFR-CNN, when tboth training and test data are drawn from the same sample space. 2) When the sample spaces of training and test data are different, SPP-CNN shows stronger generalizability than the other three predictors. 3) SPP-CNN performs prediction significantly faster than PATCHY-SAN and LFR-CNN.

Overall, the proposed SPP-CNN framework lifts the network robustness prediction to a higher level, so that the prediction tasks can be accomplished faster and more precise. This investigation reveals that the great potential of deep neural networks can be further explored for broader applications in the future.

#### REFERENCES

- [1] A.-L. Barabási, Network Science. Cambridge University Press, 2016.
- [2] M. E. Newman, *Networks: An Introduction*. Oxford University Press, 2010.
- [3] G. Chen, X. Wang, and X. Li, *Fundamentals of Complex Networks:* Models, Structures and Dynamics, 2nd ed. John Wiley & Sons, 2014.
- [4] C. M. Schneider, A. A. Moreira, J. S. Andrade, S. Havlin, and H. J. Herrmann, "Mitigation of malicious attacks on networks," *Proceedings* of the National Academy of Sciences, vol. 108, no. 10, pp. 3838–3841, 2011.
- [5] W. Ellens and R. E. Kooij, "Graph measures and network robustness," arXiv preprint arXiv:1311.5064, 2013.
- [6] S. Freitas, D. Yang, S. Kumar, H. Tong, and D. H. Chau, "Graph vulnerability and robustness: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2022, doi:10.1109/TKDE.2022.3163672 (online published).
- [7] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Controllability of complex networks," *Nature*, vol. 473, no. 7346, pp. 167–173, 2011.

- [8] Z. Z. Yuan, C. Zhao, Z. R. Di, W.-X. Wang, and Y.-C. Lai, "Exact controllability of complex networks," *Nature Communications*, vol. 4, p. 2447, 2013.
- [9] M. Pósfai, Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Effect of correlations on network controllability," *Scientific Reports*, vol. 3, p. 1067, 2013.
- [10] G. Menichetti, L. Dall'Asta, and G. Bianconi, "Network controllability is determined by the density of low in-degree and out-degree nodes," *Physical Review Letters*, vol. 113, no. 7, p. 078701, 2014.
- [11] L. Wang, X. Wang, G. Chen, and W. K. S. Tang, "Controllability of networked MIMO systems," *Automatica*, vol. 69, pp. 405–409, 2016.
- [12] L. Wang, X. Wang, and G. Chen, "Controllability of networked higherdimensional systems with one-dimensional communication channels," *Royal Society Philosophical Transactions A*, vol. 375, no. 2088, p. 20160215, 2017.
- [13] L. Xiang, F. Chen, W. Ren, and G. Chen, "Advances in network controllability," *IEEE Circuits and Systems Magazine*, vol. 19, no. 2, pp. 8–32, 2019.
- [14] J.-N. Wu, X. Li, and G. Chen, "Controllability of deep-coupling dynamical networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 12, pp. 5211–5222, 2020.
- [15] Y. Lu, Y. Zhao, F. Sun, and R. Liang, "Measuring and improving communication robustness of networks," *IEEE Communications Letters*, vol. 23, no. 12, pp. 2168–2171, 2019.
- [16] B. Mburano, W. Si, and W. X. Zheng, "A comparative study on the variants of r metric for network robustness," in *International Symposium* on Networks, Computers and Communications (ISNCC). IEEE, 2021, pp. 1–6.
- [17] P. Holme, B. J. Kim, C. N. Yoon, and S. K. Han, "Attack vulnerability of complex networks," *Physical Review E*, vol. 65, no. 5, p. 056109, 2002.
- [18] B. Shargel, H. Sayama, I. R. Epstein, and Y. Bar-Yam, "Optimization of robustness and connectivity in complex networks," *Physical Review Letters*, vol. 90, no. 6, p. 068701, 2003.
- [19] R. Cohen and S. Havlin, Complex Networks: Structure, Robustness and Function. Cambridge university press, 2010.
- [20] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Control centrality and hierarchical structure in complex networks," *PLoS One*, vol. 7, no. 9, p. e44459, 2012.
- [21] A. Bashan, Y. Berezin, S. Buldyrev, and S. Havlin, "The extreme vulnerability of interdependent spatially embedded networks," *Nature Physics*, vol. 9, pp. 667–672, 2013.
- [22] H. Chan and L. Akoglu, "Optimizing network robustness by edge rewiring: A general framework," *Data Mining and Knowledge Discov*ery, vol. 30, no. 5, pp. 1395–1425, 2016.
- [23] J. Liu, M. Zhou, S. Wang, and P. Liu, "A comparative study of network robustness measures," *Frontiers of Computer Science*, vol. 11, no. 4, pp. 568–584, 2017.
- [24] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, pp. 35–41, 1977.
- [25] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
- [26] J. Wu, B. Mauricio, Y.-J. Tan, and H.-Z. Deng, "Natural connectivity of complex networks," *Chinese Physics Letters*, vol. 27, no. 7, p. 078902, 2010.
- [27] M. Fiedler, "Algebraic onnectivity of graphs," Czechoslovak Mathematical Journal, vol. 23, no. 2, pp. 298–305, 1973.
- [28] D. J. Klein and M. Randić, "Resistance distance," *Journal of Mathematical Chemistry*, vol. 12, no. 1, pp. 81–95, 1993.
- [29] A. Gavili and X.-P. Zhang, "On the shift operator, graph frequency, and optimal filtering in graph signal processing," *IEEE Transactions on Signal Processing*, vol. 65, no. 23, pp. 6303–6318, 2017.
- [30] K. Yamashita, Y. Yasuda, R. Nakamura, and H. Ohsaki, "On the predictability of network robustness from spectral measures," in 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), vol. 2. IEEE, 2019, pp. 24–29.
- [31] A. Zeng and W. Liu, "Enhancing network robustness against malicious attacks," *Physical Review E*, vol. 85, no. 6, p. 066130, 2012.
- [32] C. M. Schneider, N. Yazdani, N. A. Araújo, S. Havlin, and H. J. Herrmann, "Towards designing robust coupled networks," *Scientific Reports*, vol. 3, no. 1, pp. 1–7, 2013.
- [33] Y. Lou, S. Xie, and G. Chen, "Searching better rewiring strategies and objective functions for stronger controllability robustness," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 6, pp. 2112–2116, 2021.

- [34] S. Wang, J. Liu, and Y. Jin, "A computationally efficient evolutionary algorithm for multiobjective network robustness optimization," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 3, pp. 419–432, 2021.
- [35] Y. Lou, L. Wang, S. Xie, and G. Chen, "Approximating the controllability robustness of directed random-graph networks against random edge-removal attacks," *International Journal of Control Automation and Systems*, 2022.
- [36] Y. Lou, Y. He, L. Wang, and G. Chen, "Predicting network controllability robustness: A convolutional neural network approach," *IEEE Transactions on Cybernetics*, vol. 52, no. 5, pp. 4052–4063, 2022.
- [37] P. Sun, R. E. Kooij, Z. He, and P. Van Mieghem, "Quantifying the robustness of network controllability," in *International Conference on System Reliability and Safety (ICSRS)*. IEEE, 2019, pp. 66–76.
- [38] P. Sun, R. E. Kooij, and P. Van Mieghem, "Reachability-based robustness of controllability in sparse communication networks," *IEEE Transactions* on Network and Service Management, vol. 18, no. 3, pp. 2764–2775, 2021.
- [39] A. Dhiman, P. Sun, and R. Kooij, "Using machine learning to quantify the robustness of network controllability," in *International Conference* on Machine Learning for Networking. Springer, 2021, pp. 19–39.
- [40] C. Fan, L. Zeng, Y. Sun, and Y.-Y. Liu, "Finding key players in complex networks through deep reinforcement learning," *Nature Machine Intelligence*, vol. 2, pp. 317–324, 2020.
- [41] M. Grassia, M. De Domenico, and G. Mangioni, "Machine learning dismantling and early-warning signals of disintegration in complex systems," *Nature Communications*, vol. 12, no. 5190, 2021.
- [42] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.
- [43] Y. Lou, Y. He, L. Wang, K. F. Tsang, and G. Chen, "Knowledgebased prediction of network controllability robustness," *IEEE Transactions on Neural Networks and Learning Systems*, 2021, doi:10.1109/TNNLS.2021.3071367 (online published).
- [44] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," arXiv preprint arXiv:1609.02907, 2016.
- [45] W. L. Hamilton, "Graph representation learning," Synthesis Lectures on Artifical Intelligence and Machine Learning, vol. 14, no. 3, pp. 1–159, 2020.
- [46] Y. Lou, R. Wu, J. Li, L. Wang, X. Li, and G. Chen, "A learning convolutional neural network approach for network robustness prediction," *arXiv preprint arXiv:2203.10552*, 2022.
- [47] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904– 1916, 2015.
- [48] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *International Conference on Machine Learning* (*ICML*), 2016, pp. 2014–2023.
- [49] C.-T. Chen, *Linear System Theory and Design*, 3rd ed. Oxford University Press, 1998.
- [50] Y. Lou, R. Wu, J. Li, L. Wang, and G. Chen, "A convolutional neural network approach to predicting network connectedness robustness," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 4, pp. 3209–3219, 2021.
- [51] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv Preprint: 1409.1556, 2014.
- [52] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.
- [53] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.
- [54] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European Conference on Computer Vision*. Springer, 2014, pp. 818–833.
- [55] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 580–587.
- [56] K. Grauman and T. Darrell, "The pyramid match kernel: Discriminative classification with sets of image features," in *IEEE International Conference on Computer Vision (ICCV)*, vol. 2. IEEE, 2005, pp. 1458–1465.
- [57] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2. IEEE, 2006, pp. 2169–2178.

- [58] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.
- [59] A.-L. Barabási, "Scale-free networks: A decade and beyond," *Science*, vol. 325, no. 5939, pp. 412–413, 2009.
- [60] Y. Lou, L. Wang, K.-F. Tsang, and G. Chen, "Towards optimal robustness of network controllability: An empirical necessary condition," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 9, pp. 3163–3174, 2020.
- [61] P. Erdös and A. Rényi, "On the strength of connectedness of a random graph," Acta Mathematica Hungarica, vol. 12, no. 1-2, pp. 261–267, 1964.
- [62] Y. Lou, L. Wang, and G. Chen, "Toward stronger robustness of network controllability: A snapback network model," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 9, pp. 2983–2991, 2018.
- [63] Y. Lou, D. Yang, L. Wang, C. Tang, and G. Chen, "Controllability robustness of henneberg-growth complex networks," *IEEE Access*, vol. 10, pp. 5103–5114, 2022.
- [64] K.-I. Goh, B. Kahng, and D. Kim, "Universal behavior of load distribution in scale-free networks," *Physical Review Letters*, vol. 87, no. 27, p. 278701, 2001.
- [65] M. E. Newman and D. J. Watts, "Renormalization group analysis of the small-world network model," *Physics Letters A*, vol. 263, no. 4-6, pp. 341–346, 1999.
- [66] W. H. Kruskal and W. A. Wallis, "Use of ranks in one-criterion variance analysis," *Journal of the American statistical Association*, vol. 47, no. 260, pp. 583–621, 1952.

## Supplementary Information for "SPP-CNN: An Efficient Framework for Network Robustness Prediction"

Chengpei Wu, Yang Lou, Lin Wang, Junli Li, Xiang Li and Guanrong Chen



Fig. S1: Boxplots of prediction errors obtained by SPP-CNN, CNN-RP, PATCHY-SAN, and LFR-CNN. Networks of  $S_1$  and  $N_a \in [700, 1300]$  are used for both training and test datasets. Connectivity robustness of directed networks under maximum-degree node attacks is predicted.



Fig. S2: Boxplots of prediction errors obtained by SPP-CNN, CNN-RP, PATCHY-SAN, and LFR-CNN. Networks of  $S_1$  and  $N_a \in [700, 1300]$  are used for both training and test datasets. Controllability robustness of directed networks under random node attacks is predicted.



Fig. S3: Boxplots of prediction errors obtained by SPP-CNN, CNN-RP, PATCHY-SAN, and LFR-CNN. Networks of  $S_1$  and  $N_a \in [700, 1300]$  are used for both training and test datasets. Connectivity robustness of undirected networks under maximum-degree node attacks is predicted.



Fig. S4: Boxplots of prediction errors obtained by SPP-CNN, CNN-RP, PATCHY-SAN, and LFR-CNN. Networks of  $S_1$  and  $N_a \in [700, 1300]$  are used for both training and test datasets. Controllability robustness of undirected networks under random node attacks is predicted.



Fig. S5: Prediction errors obtained by SPP-CNN, CNN-RP, PATCHY-SAN, and LFR-CNN for unseen network sizes (UNS). Connectivity robustness of directed networks under maximum-degree node attacks is predicted.



Fig. S6: Prediction errors obtained by SPP-CNN, CNN-RP, PATCHY-SAN, and LFR-CNN for unseen network sizes (UNS). Controllability robustness of directed networks under random node attacks is predicted.



Fig. S7: Prediction errors obtained by SPP-CNN, CNN-RP, PATCHY-SAN, and LFR-CNN for unseen network sizes (UNS). Connectivity robustness of undirected networks under maximum-degree node attacks is predicted.



Fig. S8: Prediction errors obtained by SPP-CNN, CNN-RP, PATCHY-SAN, and LFR-CNN for unseen network sizes (UNS). Controllability robustness of undirected networks under random node attacks is predicted.



Fig. S9: Numbers of superiors and inferiors obtained by SPP-CNN, compared to each one of CNN-RP, PATCHY-SAN, and LFR-CNN. Connectivity robustness of directed networks under maximum-degree node attacks is predicted.



Fig. S10: Numbers of superiors and inferiors obtained by SPP-CNN, compared to each one of CNN-RP, PATCHY-SAN, and LFR-CNN. Controllability robustness of directed networks under random node attacks is predicted.



Fig. S11: Numbers of superiors and inferiors obtained by SPP-CNN, compared to each one of CNN-RP, PATCHY-SAN, and LFR-CNN. Connectivity robustness of undirected networks under maximum-degree node attacks is predicted.



Fig. S12: Numbers of superiors and inferiors obtained by SPP-CNN, compared to each one of CNN-RP, PATCHY-SAN, and LFR-CNN. Controllability robustness of undirected networks under random node attacks is predicted.





(a) Connectivity robustness of directed networks under maximum-degree node attacks.



(c) Connectivity robustness of undirected networks under maximum-degree node attacks.

(b) Connectivity robustness of directed networks under random node attacks.



(d) Controllability robustness of undirected networks under random node attacks.

Fig. S13: Prediction errors obtained by SPP-CNN, CNN-RP, PATCHY-SAN, and LFR-CNN for unseen network topology (UNT).



(a) Connectivity robustness of directed networks under maximum-degree node attacks.





(b) Controllability robustness of directed networks under random node attacks.



<sup>(</sup>d) Controllability robustness of undirected networks under random node attacks.

Fig. S14: Numbers of superiors and inferiors obtained by SPP-CNN, compared to each one of CNN-RP, PATCHY-SAN, and LFR-CNN for unseen network topology (UNT).