Fast Cascading Outage Screening based on Deep Convolutional Neural Network and Depth-First Search

Yan Du, *Student Member, IEEE*, Fangxing Li, *Fellow, IEEE,*
Tongxin Zheng, *Senior Member IEEE*, Jiang Li, *Senior Member, IEEE*

***Abstract*—In this paper, a data-driven method is proposed for fast cascading outage screening in power systems. The proposed method combines a deep convolutional neural network (deep CNN) and a depth-first search (DFS) algorithm. First, a deep CNN is constructed as a security assessment tool to evaluate system security status based on observable information. With its automatic feature extraction ability and the high generalization, a well-trained deep CNN can produce estimated AC optimal power flow (ACOPF) results for various uncertain operation scenarios, i.e. fluctuated load and system topology change, in a nearly computation-free manner. Second, a scenario tree is built to represent the potential operation scenarios and the associated cascading outages. The DFS algorithm is developed as a fast screening tool to calculate the expected security index value for each cascading outage path along the entire tree, which can be a reference for system operators to take predictive measures against system collapse. The simulation results of applying the proposed deep CNN and the DFS algorithm on standard test cases verify that their accuracy and computational efficiency is thousands of times faster than the model-based traditional approach, which implies the great potential of the proposed algorithm for online applications.**

***Keywords*—Cascading outage, deep convolutional neural network (deep CNN), depth-first search (DFS), scenario tree, security assessment.**

NOMENCLATURE

Indices, acronyms, and parameters

v_i	Voltage magnitude of the i^{th} bus (p.u.)
V_i^b	Base voltage magnitude (p.u.)
A_i^u/A_i^l	The upper and lower boundary of the voltage alarm limit (p.u.)
S_i^u/S_i^l	The upper and lower boundary of the voltage security limit (p.u.)
P_l	The power flow of the l^{th} line (MW)
$A_{p,l}$	The alarm limit of the line flow
$S_{p,l}$	The secure limit of the line flow
SI	Security index
P_d/Q_d	Bus active/reactive load (MW)
P_g/Q_g	Generator active/reactive output (MW)
g_{ij}/b_{ij}	Line conductance/susceptance between bus i and bus j (p.u.)
θ_i	Voltage angle difference between bus i and bus j (rad)
F_{ij}	Active power flow on transmission line ij (MW)
$\omega(u,v)$	Weight value in the convolutional filter
b	Bias of the convolutional filter
N_s	Number of training samples
n	Number of buses
Conv	Convolutional layer
ReLU	Rectified linear unit
FC	Fully-connected layer
p_l	Line failure probability
T	System topology
SI_{exp}	Accumulative security index

I. INTRODUCTION

A. Motivation

Protecting the bulk power system against cascading outages is a crucial measure towards enhancing system-wide operation economy and resilience. According to the NERC definition [1], cascading outage refers to a situation where the system uncontrollably and successively loses elements triggered by an initial incident at any location. A cascading outage will result in widespread electric service interruption, which cannot be restrained from sequentially spreading beyond an area predetermined by studies. However, the growing penetration of uncertainties into the bulk power system is increasing system vulnerability, as well as the chance for cascading outages. Although the probability of a cascading outage inducing blackouts is tiny, the consequences would be catastrophic, resulting in tremendous economic losses and social impacts.

There have been several large-scale blackouts caused by cascading outages in recent years, such as the western U.S. blackout in 1996 [2], the U.S.-Canadian blackout in 2003 [3] and the Arizona-California blackout in 2011 [4]. Given the costly effects of cascading outages, NERC has required that each Transmission Planner and Planning Coordinator shall define the criteria or methodology used in the analysis to identify system instability for cascading or uncontrolled islanding during planning assessment studies [5].

In the above context, both research communities and the industries have devoted substantial efforts to studying cascading outages. However, the majority of the existing studies are founded on the conventional model-based method for cascading outage analysis, which suffers from certain computational limitations. It is mainly motivated by this consideration that in this paper we propose a data-driven method that combines the deep convolutional neural network (deep CNN) with a depth-first search (DFS) algorithm for a fast cascading outage screening and risk assessment, which aims at potential online applications under uncertain scenarios. A more

This work was financially supported in part by the CURENT research center, ISO New England, and NSF grant ECCS-1809458. Corresponding author: F. Li, e-mail: fli6@utk.edu.

Y. Du and F. Li are with the University of Tennessee, Knoxville, TN.

T. Zheng is with ISO New England, Holyoke, MA.

J. Li is with Old Dominion University, Norfolk, VA.

detailed literature review and the contributions of our work will be presented in the following subsections.

B. Literature Review

The existing research regarding cascading outages can be classified into three main categories: cascading outage simulation and pattern recognition, system vulnerability detection and risk assessment, and post-outage recovery.

In regard to the first category, simulation models have been developed to study the impact of cascading outages, including the OPA model and its multiple improved versions [6,7], the Manchester model [8], and the CASCADE model [9]. Multi-timescale cascading outage models to study both slow dynamics like thermal transient and fast dynamics like electrical instability are further developed in [10,11]. A sequential importance sampling strategy to reduce the number of cascading failures, while still capturing very rare events is proposed in [12]. To gain better statistical insights to the pattern of cascading outage propagation, [13,14] apply a Markov chain approach, where transition probabilities are estimated from historical data; while [15,16] utilize the expectation maximization method, in which the parameters of the probability function are their maximum likelihood estimates.

With respect to system vulnerability detection and risk assessment, a forward-backward Markovian tree search algorithm is introduced in [17], where the risk of the current outage is the expected risk of all its following outages. Based on this work, [18] further considers weather impacts on line outage probabilities and system risks, and it develops an associated analytical probability model. Ref. [19] studies the quantitative relationship between component failure probability and blackout risks during cascading outages, which can be used as an effective risk assessment tool under the system components change. Ref. [20] shows that cascading outage risk can be underestimated if the multiple solutions of DC optimal power flow (DCOPF) models are not considered, and then proposes remedial measures. A fast screening method for vulnerable transmission lines based on PageRank algorithm is proposed in [21], where the vulnerability degree of each line is calculated based on its post-contingency flow under the N-1 contingency of all the other lines. A branch loading assessment index is defined in [22], where a cascading fault graph based on the proposed index is designed to demonstrate the vulnerability of each transmission line.

For post-outage recovery measures, a simulation-based optimization method [23], a multi-agent system method [24], and a Markovian tree search method [25] are introduced to reduce the risk mitigation cost through generator re-dispatch and transmission capacity allocation.

The above concern motivates the development of the data-driven method as a meaningful alternative for fast cascading outage screening. As opposed to the model-based method, the data-driven method formulates an approximate mapping between the input and the output. Once the algorithm is well-trained, it is a generalized model that can automatically produce outputs from unseen new inputs, without a massive amount of analytical computation. Therefore, the data-driven method is promising with regard to future online cascading outage analyses with real-time data input.

The application of the data-driven method in cascading outage analysis is still in its initial stage in the literature. Some previous works have been dedicated to utilizing machine learning methods such as artificial neural networks [26], convolutional neural network [27] and deep autoencoder [28], for security assessment under contingency, but not really the cascading outage effects. In other words, few studies have considered the risk of the following cascading outages after a contingency event, which may cause the violation of NERC security standards. In [29], a three-stage decision tree method is proposed to classify the severity level of a cascading blackout. The system states obtained from a wide area measurement system (WAMS) are used to train the decision trees, which prove to have a high classification accuracy. In [30], the authors propose a Monte Carlo cascading failure simulation method utilizing the existing model-based software package and a risk assessment method of cascade paths based on de-correlated neural network ensembles. However, in this last work, the line flow is used as input to the neural network for system risk evaluation, which implies that power flow calculation is still needed for new test cases. The ultimate goal of the data-driven method is to utilize direct system observations, i.e., topologies, as the input to the algorithm without any additional analytical calculation for indirect measurements (such as line flow) to realize a nearly computation-free manner. Otherwise, the data-driven method can still be computation-inefficient for online applications under uncertainties.

C. Contributions

Based on the previous works, in this paper, we also propose a novel data-driven method for fast cascading outage screening and risk assessment. The proposed method is a combination of a deep convolutional neural network (deep CNN) and a depth-first search (DFS) algorithm. First, the deep CNN is constructed as a regression tool of the AC optimal power flow (ACOPF) model to quickly obtain system state variables. The state variables are then utilized to calculate a security index for evaluating outage severity. Secondly, a scenario tree is built to represent all the potential cascading paths in real-time uncertain scenarios. Also, a DFS algorithm is utilized to screen all of the cascading outage paths in the scenario tree to detect the severest path. The detection is based on the estimated security index value from the deep CNN. The screening results can serve as a reference for system operators to take corrective measures against system collapse. The main contributions of our paper are summarized as follows:

1) We propose the deep CNN as an efficient regression method for approximating ACOPF calculation. Unlike other data-driven methods that rely on system state variables as input, a well-trained deep CNN only needs direct observations, e.g., system topology and bus power injection, and will automatically generate the state variables for evaluating outage severity. Hence, it can be directly applied to new test cases without the computationally intensive power flow calculation.

2) We establish a multi-scenario tree as an efficient representation of all the potential cascading outage paths with uncertainties involved. Furthermore, we apply the DFS method for a fast cascading outage screening over the entire tree. The DFS method aims to calculate the expected accumulative security index of each cascading outage path for evaluating their severity. With a proper screening order of all the cascading

outages, the proposed DFS method can complete the traversal with extremely low elapsed time, which is highly applicable in the case of large-scale power system cascading outage screening.

The rest of the paper is organized as follows: Section II briefly introduces a security index for evaluating outage severity; Section III demonstrates the design of the proposed deep CNN for ACOPF regression; Section IV explains the construction of the scenario tree and the details of the DFS algorithm; Section V verifies the proposed deep CNN and DFS algorithm for cascading outage screening on standard test cases; finally, Section VI concludes the paper.

II. Composite Security Index for Security Assessment

To accurately evaluate the security status of a power system, a composite security index is first introduced, which measures both bus voltage limit violation and line flow violation. For each measurement, two types of limits are defined, the security limit and the alarm limit. Security limit refers to the maximum allowed range for the bus voltage and line flow, and alarm limit indicates the closeness of the system to the violation limit. Accordingly, the system security status can be categorized as one of three types: secure, alarm, and insecure. A system is in the alarm state if at least one of the measurements violates the alarm limit but is still within the security limit. A system is insecure if at least one of the measurements violates the security limit [31]. Several other measures need to be defined before proceeding to calculate the composite security index.

For bus voltage, the normalized deviation of bus voltage from the alarm limits is defined as follows:

$$d^u_{v,i} = \begin{cases} \dfrac{\left|v_i - A^u_i\right|}{V^b_i}, & if\ v_i > A^u_i \\ 0, & if\ v_i \le A^u_i \end{cases} \quad d^l_{v,i} = \begin{cases} \dfrac{\left|A^l_i - v_i\right|}{V^b_i}, & if\ v_i < A^l_i \\ 0, & if\ v_i \ge A^l_i \end{cases} \tag{1}$$

In Eq. (1), v_i is the voltage magnitude of the i^{th} bus; V^b_i is the base voltage magnitude; A^u_i and A^l_i are the upper and lower boundary of the voltage alarm limit. The normalized deviation of the alarm limit from the secure limit is defined as follows:

$$g^u_i = \frac{\left|S^u_i - A^u_i\right|}{V^b_i},\ g^l_i = \frac{\left|S^l_i - A^l_i\right|}{V^b_i} \tag{2}$$

In Eq. (2), S^u_i and S^l_i are the upper and lower boundary of the voltage security limit. For line flow, only the upper boundaries of the alarm limit and the secure limit are needed. The normalized line flow violation of the alarm limit is defined as follows:

$$\begin{aligned} d_{p,l} &= \frac{\left|\left|P_l\right| - A_{p,l}\right|}{Base\ MVA},\ if\ \left|P_l\right| > A_{p,l} \\ d_{p,l} &= 0, \qquad\qquad\quad if\ \left|P_l\right| \le A_{p,l} \end{aligned} \tag{3}$$

In Eq. (3), P_l is the power flow of the l^{th} line; $A_{p,l}$ is the alarm limit of the line flow. The normalized deviation of alarm limit from security limit is defined as follows:

$$g_{p,l} = \frac{\left|S_{p,l} - A_{p,l}\right|}{Base\ MVA} \tag{4}$$

In Eq. (4), $S_{p,l}$ is the security limit of the line flow. Based on the above definitions, the composite security index for the system is defined as follows [26]:

$$SI = \left[\sum_i \left(\frac{d^u_{v,i}}{g^u_{v,i}}\right)^{2m} + \sum_i \left(\frac{d^l_{v,i}}{g^l_{v,i}}\right)^{2m} + \sum_l \left(\frac{d_{p,l}}{g_{p,l}}\right)^{2m}\right]^{\frac{1}{2m}} \tag{5}$$

Eq. (5) is based on the concept of a hyper-ellipse inscribed within a hyper-box for measuring limit violation [32], where m is the exponent used in the hyper ellipse equation. In this study, m is set to 1. A higher value security index means that the system is at a higher risk level. For example, if both voltage magnitudes and line flows are within the alarm limit, which means that the system is operating within the secure region, then $d^u_{v,i}$, $d^l_{v,i}$, and $d_{p,l}$ are all zeroes, which leads to a zero SI; if any voltage magnitude or line flow is out of the alarm limit but still within the security limit, which means that the system can maintain operation for a short time, then d is smaller than g, which leads to a value security index that is larger than 0, but mostly below 1; and finally, if any voltage magnitude or line flow is above the security limit, which means that the system is close to collapse, then d will be larger than g, which will definitely lead to a security index larger than 1.

III. Deep CNN-based Security Assessment

A. A brief on deep CNN

A deep CNN is a type of artificial neural network with multiple hidden layers, and is known for its strong capabilities in processing data that has a grid-like topology, e.g., image data. Images are represented by a 2-D matrix with pixels filled in. The crux of a deep CNN lies in that it formulates a hierarchical structure that mimics the visual cortex of humans. According to visual neuroscience, in image recognition, our brain first perceives the color and brightness of the observed object, then the edge, angle, line, and other local details, followed by the shape, texture and more abstract information, and finally the entire image.

The convolutional neural network follows the logic of the visual cortex. It consists of multiple convolutional layers, each of which contains several convolution kernels. Each convolution kernel scans the entire input to capture the detailed local features. All the captured features will formulate a feature map for the neural network to identify. As the convolutional layer goes deeper, more high-order and abstract features will be captured, preserving the most useful information for image recognition.

Deep CNNs have an important feature, which is sparse connectivity [33]. In conventional neural networks, usually every output unit is connected to every input unit. The number of connection parameters that need to be trained can be tremendous. In the case of the deep CNN, each output unit in the feature map is only connected to a square patch, named as field of review, from the input that is closest to its location, instead of the entire input. This is called sparse connectivity. The reason for doing so is that in one image, one pixel is closely related to its neighboring pixels, but is less related to the pixels in the farther distance. Hence the connections between the less related units are removed. With sparse connectivity, the number of parameters for training is greatly reduced, which improves computational efficiency.

B. Mapping power grid data to deep CNN input data

A deep CNN is a natural fit for solving power system problems for the following two reasons [34,35]:

1) The power system topology has a grid-like structure, which can be described by matrices such as the nodal admittance matrix, the element-bus incidence matrix, and the branch-path incidence matrix.

2) The power system possesses the feature of sparse connectivity: for instance, the voltage level at one bus is closely related to its neighboring buses, while it is less affected by the buses that are far away.

Consequently, a hierarchical deep CNN will learn the element-bus relationship, the line connection, and the entire power system topology layer by layer. The term "deep" indicates that the proposed neural network contains multiple hidden layers to fully capture the features of power system raw data.

In the case of system security assessment, the function of the deep CNN is to approximate the ACOPF calculation and to obtain system state variables. The state variables can then be used to calculate the security index to evaluate system security status. To achieve this function, the first step is to map power system raw data to a grid-like structure for the CNN to analyze.

The ACOPF model is shown as follows:

$$\min \sum_{g=1}^{N_g} C_P(P_g)+C_Q(Q_g) \tag{6}$$

$$s.t. \sum_{g\in i} P_g - \sum_{d\in i} P_d = v_i \sum_{j\in i} v_j (g_{ij}\cos\theta_{ij} + b_{ij}\sin\theta_{ij}) \tag{7}$$

$$s.t. \sum_{g\in i} Q_g - \sum_{d\in i} Q_d = v_i \sum_{j\in i} v_j (g_{ij}\sin\theta_{ij} - b_{ij}\cos\theta_{ij}) \tag{8}$$

$$-F_{ij,\max} \le \left|F_{ij}\right| \le F_{ij,\max} \tag{9}$$

$$v_{i,\min} \le v_i \le v_{i,\max} \tag{10}$$

$$P_{g,\min} \le P_g \le P_{g,\max},\ Q_{g,\min} \le Q_g \le Q_{g,\max} \tag{11}$$

In Eq. (6)-(11), the known parameters are the bus active/reactive load P_d, Q_d, and system topology g_{ij}, b_{ij}, which will be the input to the deep CNN; and the unknowns are the bus active/reactive generation, P_g, Q_g, bus voltage magnitude v_i, and bus voltage angle θ_i. Given that we only need bus voltage for the security index calculation, the deep CNN will only output v_i and θ_i in this case. However, notice that the input parameters differ in their dimension. Given an n-bus power system, the P_d and Q_d will be both 1×n vectors, while g_{ij} and b_{ij} are both in an n×n matrix. The deep CNN requires that the input known quantities should have the same dimensions. For example, for the image data, each image has the following dimensions: w (width) ×h (height) ×c (number of color channels), where the dimension for each color channel is the same. To reach this requirement, we utilize the following 1× n vector to represent system topology:

$$\text{diag}(\text{imag}(\text{Y})) = \text{diag}(\text{B}) = \begin{bmatrix} b_{11} & b_{22} & \cdots \end{bmatrix} \tag{12}$$

In Eq. (12), Y is the bus admittance matrix, and B is the bus susceptance matrix. The reason for utilizing bus self-susceptance elements to represent system topology change is that whenever there is a line outage, the self-susceptance elements will definitely change, but not necessarily the self-conductance elements, since some lines have zero resistance. By removing the non-diagonal elements in the B matrix, we only keep the most dominant elements as an efficient representation of system topology. Since deep CNN regression is a data-driven method, the regression error caused by the missing data in the G matrix and the B matrix will be automatically made up via iterative training based on existing data samples. With the above simplification, a deep CNN regression for ACOPF calculation only requires three 1×n vectors as the input. The volume of training data is acceptable even in case of large-scale power systems.

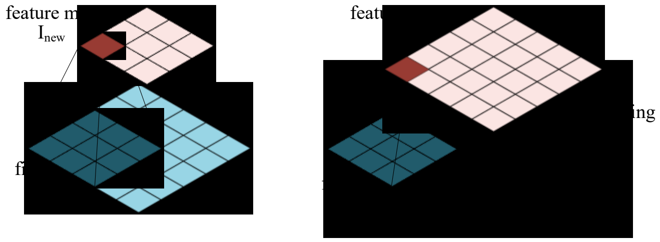
C. Constructing the deep CNN

1) An illustration of convolution operation

In a convolutional neural network, the core component is the convolutional layer. A convolutional layer is composed of trainable convolution kernels, or the filter. The function of the filter is to extract features from the input to generate feature maps that are representatives of the input. The feature extraction can be mathematically expressed as follows:

$$I_{new}(i,j) = \sum_{u=0}^{c-1}\sum_{v=0}^{c-1} I(u+i, v+j)\cdot\omega(u,v)+b \tag{13}$$

In Eq. (13), $I_{new}(i,j)$ is a single unit in the newly generated feature map I from the convolutional layer; $I(u,v)$ is a single unit in the original input; $\omega(u,v)$ is a single unit in the filter, which is also called the weight parameter; c is the size of the filter; b is the bias. Fig. 1. gives an illustrative example of the above convolution operation:



(a) no padding (b) with padding

Fig. 1. Illustration of convolution operation

In Fig. 1(a), the size of the input is 5×5, and the size of the filter is 3×3. Each unit in the feature map is the weighted sum of 9 units in the input. The filter scans the input with a step size of 1, and hence the size of the feature map is 3×3. The feature map thus contains the aggregated information from the input. If we want to retain the size of the input, a padding method can be used, as shown in Fig. 1 (b). Two additional rows and columns are added to the input with 0 filled, this is called zero-padding. Then after the convolution operation, the feature map will have the same size as the input.

Following the explanation above, it can be observed that the feature extraction function of the convolutional layer refers to adding different weights to the inputs. In conventional machine learning, usually the features of the input have to be computed and selected manually to feed to the neural network for the algorithm to learn. In the case of the deep CNN, because of the existence of multiple hidden layers and multiple filters in each layer, the filters automatically capture the features of the input by changing the weights. After the deep CNN is well trained, the weights of the filters will have been properly selected so that the most obvious features from the input will have larger weights, while the less important features are neglected. In this way, the desired output can be obtained. In a convolutional layer, multiple filters usually exist, and each filter will generate a different feature map. The purpose of utilizing multiple filters is to observe the input from different perspectives, i.e. assigning different weights to the same input unit, so that a comprehensive feature extraction can be obtained. The above explains the automatic feature extraction ability of the deep CNN.

2) Back-propagation algorithm

In Eq. (13), the weight parameter ω and the bias b are the unknown variables. The values of the unknown variables are obtained via a back-propagation algorithm. To begin, a loss function is defined to describe the accuracy of the output from the neural network. A lower loss value indicates higher accuracy of the model. In the static security assessment problem, mean square error is used as the loss function:

$$L=\frac{1}{N_S}\sum_{s=1}^{N_S}\Big(\frac{1}{n}\sum_{i=1}^{n}(\theta_{i,s}^{*}-\theta_{i,s})^2+\frac{1}{n}\sum_{i=1}^{n}(v_{i,s}^{*}-v_{i,s})^2+(SI_s^{*}-SI_s)^2\Big) \qquad (14)$$

In Eq. (14), N_S is the number of training samples, n is the number of power system buses, $\theta_{i,s}^{*}$ and $v_{i,s}^{*}$ are the desired output from the deep CNN, i.e., the actual bus voltage angle and bus voltage magnitude, $\theta_{i,s}$ and $v_{i,s}$ are the estimated bus voltage angle and bus voltage magnitude. Since we need to evaluate the system security status, a third term is added to the loss function, which is the difference between the actual security index value SI_s^{*} and the estimated security index value SI_s. As can be seen, the objective of the deep CNN is to minimize the deviation between the estimation and the ground truth to formulate an accurate enough ACOPF regression model.

Furthermore, to avoid the issue of overfitting, which is a common problem in regression analysis due to the existence of abnormal values, we add L_2 regularization to the loss function (14). Generalization means that the well-trained neural network can be effective across a wide range of inputs, not just the training data that has been fed to the neural network for learning. Sometimes a deep CNN can grow very complex with large values as its weights and biases, where instead of understanding the data, the deep CNN will memorize the one-to-one mapping between the input and the output, which leads to the result that the deep CNN fits well on the training set, but performs poorly on the test set. This is because all the data in the test set are unknown to the deep CNN, and it has no memorized information for the new samples. The above problem is called overfitting.

L_2 regularization is a widely used method to avoid the issue of overfitting. L_2 regularization refers to a norm-2 penalty of weight parameters, as shown in Eq. (15):

$$L_{reg}=L+\frac{\alpha}{2}\boldsymbol{\omega}^{T}\boldsymbol{\omega} \qquad (15)$$

In Eq. (15), α is called the regularization parameter, which is a positive number. The penalty term $\boldsymbol{\omega}^{\boldsymbol{T}}\boldsymbol{\omega}/2$ stands for model complexity. An overfitted model that intends to match all the input samples, including abnormal values and noises, will have higher model complexity. By adding the penalty term to the loss function, the value of weight parameters will be decreased, and the model will evolve toward low complexity and high generalization.

Upon the calculation of the loss function, the weight and bias are updated based on the first partial derivatives:

$$\omega_l^{(k+1)}=\omega_l^{(k)}-\eta\,\frac{\partial L_{reg}}{\partial J_{N_L-1}^{(k)}}\cdot\frac{\partial J_{N_L-1}^{(k)}}{\partial J_{N_L-2}^{(k)}}\cdot\cdots\frac{\partial J^{(k)}}{\partial \omega_l} \qquad (16)$$

In Eq. (16), k is the index of iteration, l is the index of the convolutional layer, N_L is the total number of convolutional layers, $J_l^{(k)}$ is the output of the l^{th} layer, η is called the learning rate. Since the deep CNN has numerous convolutional layers, the chain rule is applied to calculate the partial derivative of the parameters at each layer. The bias b is updated similarly. As can be observed, the back-propagation algorithm is essentially a gradient descent search method. The word "back-propagation" means that the neural network parameters are updated from the last layer to the first layer based on the difference between the actual output and the desired output.

3) Design of deep CNN structure

The structure of the deep CNN is demonstrated in Fig. 2 (see next page). It consists of two convolutional (Conv) layers and five fully-connected (FC) layers. The functions of these deep CNN layers are explained in detail as follows:

a) The input data is a 3×*n* matrix, where *n* is the number of buses. The 3×*n* data correspond to three 1×*n* vectors, i.e., the real loads of *n* buses, the reactive loads of *n* buses, and the *n* diagonal elements of the B matrix.

The first convolutional layer has a filter with the size of [3,3,1,12], where the first three numbers are the height, width, and the depth of the filter. The last figure is the number of filters. In this layer, 12 filters will be sampling the input data. As a result, the input data is deepened after being scanned by the filter. In addition, zero-padding is applied to maintain the original size of the input data. Hence the output of the first convolutional layer has the size of [3,*n*,12].

The filter has a size of 3×3, which means that it assumes the three neighboring buses have strong interrelations, e.g., bus 2-4, bus 3-5, since each time the filter samples a size of 3×3 from the input. This is in accordance with physical laws because the bus voltage angle is mainly affected by its closest neighboring buses. The size of the filter can also be increased to include additional neighboring buses, but this comes with a larger quantity of parameters that need to be trained.

b) The output from the first convolutional layer goes through an activation function. The activation function adds nonlinearity to the feature extraction. This is because Eq. (13) is a linear transformation. However, the ACOPF model (6)-(11) is nonlinear and nonconvex. Introducing the activation function to feature extraction removes the limitation of linear representation.

In this study we utilize a rectified linear unit (ReLU) as the activation function. The ReLU has the following mathematical expression: $f(x)=\max(x,0)$. The quasi-linearity feature of the ReLU makes it derivable and thus applicable to the gradient descent search during neural network training.

c) The output from the ReLU function goes through the next convolutional layer, which has filters with the size of [3,3,12,24]. More features are extracted by the second convolutional layer.

d) The output from the second convolutional layer has a size of [3, *n*, 24], which is a 3-D tensor. It is further flattened as a 1 ×(3×*n*×24) vector and goes through a FC layer, FC1. In FC1, there is a connection between each neuron and each element in the input. In this case, the size of

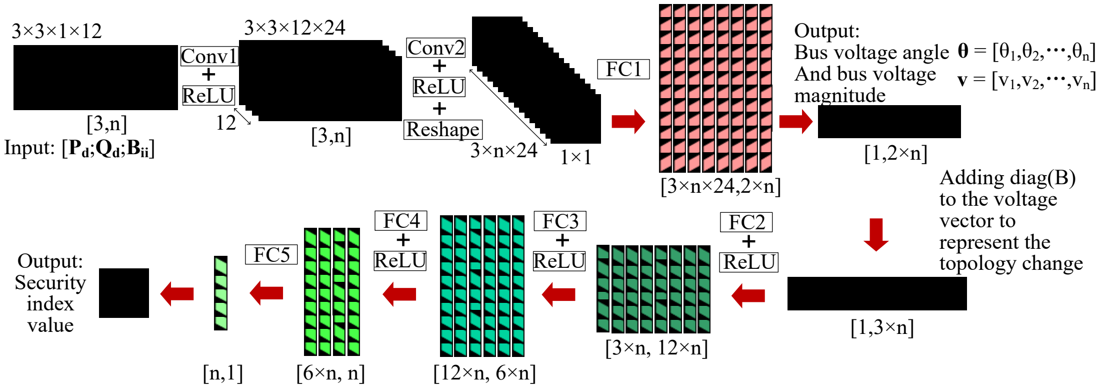


Fig. 2. Deep CNN structure for security assessment

the weight parameters in the FC1 layer is [3×n×24, 2×n], and the size of the bias is 2×n. After matrix multiplication, the output will become a vector with the size of [1, 2×n], which is a combination of n bus voltage magnitudes and n bus voltage angles.

e) Because we need to evaluate the system security status, the obtained voltage variables will further go through the next four FC layers to calculate the security index, which is a regression of Eq. (5). Before sending the voltage variables to the FC layer, the diagonal elements of B matrix are added to the voltage tensor to reflect the system topology change. This is because in Eq. (5), the line flow is related to system topology.

The four following FC layers, FC2 to FC5 in Fig. 2, have sizes of [3×n, 12×n], [12×n, 6×n], [6×n, n], and [n, 1], respectively. After matrix multiplication, the final output will be a 1×1 scalar, which is the security index value.

Via the deep CNN described above, both the system state variables and system security index can be obtained. Some may argue that since we only need the security index to evaluate system status, there is no need to output the bus voltage variable, which may result in a less complicated neural network structure. However, the security index value only shows the system security status as a whole, while it cannot reflect local weaknesses and vulnerabilities. With system state variables, we can gain insights into the local voltage margin and line flow margin. In summary, the state variables cover more detailed information on system operation than the security index value.

D. Training sample generation

In the training phase of the deep CNN, large quantities of training samples are required for fine-tuning the neural network parameter. Since the proposed deep CNN is aimed for cascading outage analysis, in the training sample, power flow results for k outage stages are included, where k indicates the number of electrical components that are out of service. In this study, we mainly consider line outage contingency. During power system operation, once a transmission line is tripped, it may cause overloading of other transmission lines and induce cascading line outages. The probability of the l^{th} transmission line failure is calculated as follows [30]:

$$p_l = \begin{cases} 1, |P_l| \ge S_{p,l} \\ \dfrac{|P_l| - A_{p,l}}{S_{p,l} - A_{p,l}}, & A_{p,l} \le |P_l| \le S_{p,l} \end{cases} \quad (17)$$

At each outage stage, based on Eq. (17), the line with the highest failure probability is selected as the tripped line.

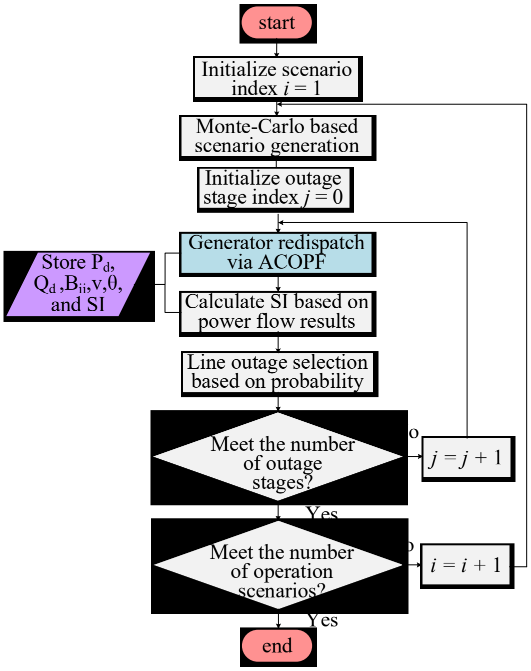

start
Initialize scenario index i = 1
Monte-Carlo based scenario generation
Initialize outage stage index j = 0
Generator redispatch via ACOPF
Store Pd, Qd ,Bii,v,θ, and SI
Calculate SI based on power flow results
Line outage selection based on probability
Meet the number of outage stages?
o
j = j + 1
Yes
Meet the number of operation scenarios?
o
i = i + 1
Yes
end

Fig. 3. Flowchart of generating cascading outage training samples

The whole process of generating cascading contingency training samples is shown in Fig. 3, and is explained as follows:

1) To begin, an operation scenario is randomly generated based on Monte-Carlo simulation to represent real-time uncertainties. In this study, we mainly consider load variations;

2) Under the generated scenario, the model-based ACOPF is conducted to evaluate system security status; the system parameters and power flow results are stored for future training of the deep CNN;

3) Based on the obtained power flow results, the tripped line is selected according to Eq. (17). If there are several lines that are out of limit, the line with the highest probability is selected as the tripped line;

4) Since we consider cascading outages in this study, if the number of line outage stages reaches k, then go to step 5); else go back to steps 2)-3) to repeat the above process;

5) If enough operation scenarios have been generated, then the whole process is complete; else go back to step 1) to regenerate operation scenarios and repeat the above cascading outage process.

IV. Cascading Outage Screening based on Depth-First Search Algorithm

In the previous section, a deep CNN is constructed to approximate ACOPF for evaluating system security status. In this section, we will demonstrate how to apply the calculated security index in cascading outage screening under multiple real-time scenarios.

Given that the cascading outage is a time sequential process, we construct a scenario tree to represent the continuous dynamic changes of the system operation scenarios, which is shown in Fig. 4 [36].

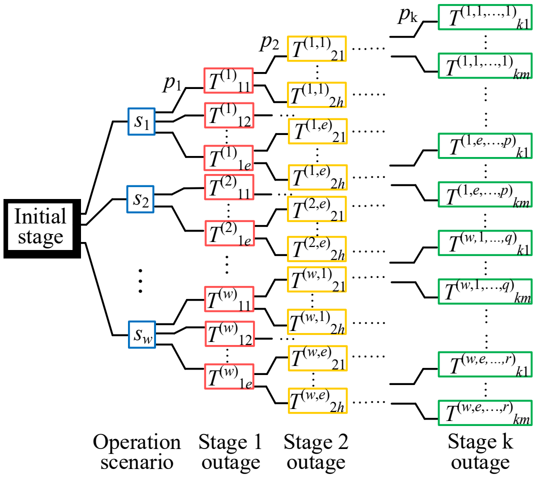


Fig. 4. Multi-scenario tree for cascading outage screening

Fig. 4 corresponds with the process of training the sample generation shown in Fig. 3. In Fig. 4, beginning at the initial stage, different operation scenarios are first generated to represent real-time uncertainties using Monte Carlo simulation. The uncertainties are regarded as disturbances that trigger the following cascading line outages. At each tree node, i.e., at each outage stage, T stands for system topology, the superscript records all the previous stages, and the subscript indicates the current stage. Take $\mathrm{T}^{(1,1)}_{21}$ as an example. In the superscript "(1,1)" , the first "1" indicates the operation scenario 1, and the second "1" indicates the first line outage scenario in 1st outage stage. In the subscript "21", the "2" indicates the 2nd outage stage, and the "1" indicates the first line outage scenario in the 2nd outage stage.

On each branch that connects two tree nodes, p_k is the line failure probability, which can be calculated by Eq. (17). A cascading outage path is defined as a path that starts from the initial stage and terminates at the k^{th} outage stage. A value is assigned to each node along the path, namely the security index *SI*. The goal of cascading outage screening is to evaluate the severity of each cascading outage path based on *SI*.

We define the following accumulative security index for severity measurement:

$$\begin{aligned} &SI^{(k)}_{\exp}=p_k SI^{(k)} \\ &SI^{(i-1)}_{\exp}=p_{i-1}(SI^{(i-1)}+SI^{(i)}_{\exp}),\ for\ i=k,k-1,\ldots,2 \end{aligned} \tag{18}$$

In Eq. (18), starting from the k^{th} outage stage, the accumulative security index is calculated in a recursive manner. For example, for the cascading outage path $s_1 \rightarrow T^{(1)}_{11} \rightarrow T^{(1,1)}_{21} \rightarrow \ldots\ldots \rightarrow T^{(1,1,\ldots,1)}_{k1}$, the accumulative security index is calculated as follows:

$$\begin{aligned} &T^{(1,1,\ldots,1)}_{k1}:\ SI^{(k)}_{\exp}=p_k SI^{(k)} \\ &T^{(1,\ldots,1)}_{(k-1)1}:\ SI^{(k-1)}_{\exp}=p_{k-1}(SI^{(k-1)}+SI^{(k)}_{\exp}) \\ &\qquad\qquad \cdots\cdots \\ &T^{(1,1)}_{21}:\ SI^{(2)}_{\exp}=p_2(SI^{(2)}+SI^{(3)}_{\exp}) \\ &T^{(1)}_{11}:\ SI^{(1)}_{\exp}=p_1(SI^{(1)}+SI^{(2)}_{\exp}) \end{aligned} \tag{19}$$

Finally, $SI^{(1)}_{\exp}$ is taken as the final accumulative value of the entire cascading outage path.

Based on Eq. (18), we design the following depth-first search (DFS) algorithm for calculating the accumulative security index for each cascading outage path, as shown in Fig. 5. The main idea of the DFS algorithm is to first explore the cascading outage stages along one path as deeply as possible until reaching the last outage stage, while storing the order of line outages and the associated security index; then to backtrack to the previous outage stages and update their expected security index. If all of the line outages at one outage stage have been scanned, return to the previous outage stage and switch to another line outage as the source node and repeat the above process until all of the cascading outage paths have been screened. The DFS algorithm is a natural fit for cascading outage screening because its forward-backward propagation corresponds with the recursive calculation of $SI^{(k)}_{\exp}$ in Eq. (18).

Note that in the above process, the original security index at each outage stage has already been calculated by the deep CNN. Once the deep CNN is well trained, it can be directly applied to new test cases in the multi-scenario tree and automatically generate ACOPF results and the associated security index, greatly reducing computational burden. In the next section, the simulation studies prove that the combination of the deep CNN and the above DFS algorithm makes it possible to scan a large-scale multi-scenario tree with extremely low time cost, while maintaining the desired accuracy.

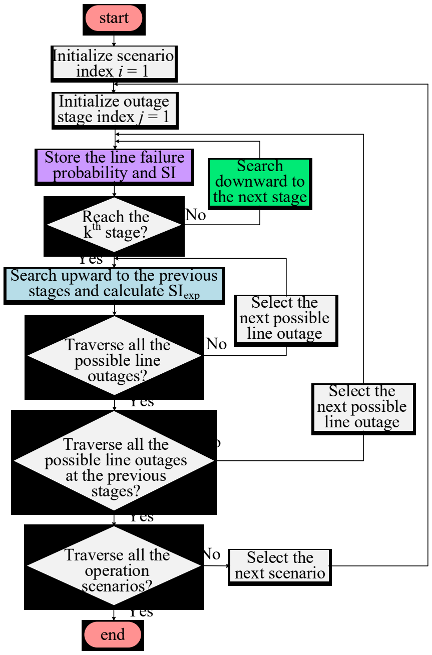

start
Initialize scenario index i = 1
Initialize outage stage index j = 1
Store the line failure probability and SI
Search downward to the next stage
Reach the k^th stage?
No
Yes
Search upward to the previous stages and calculate SI_exp
Select the next possible line outage
Traverse all the possible line outages?
No
Yes
Select the next possible line outage
Traverse all the possible line outages at the previous stages?
No
Yes
Traverse all the operation scenarios?
No
Select the next scenario
Yes
end

Fig. 5. Flow chart of DFS algorithm

V. Simulation Analysis

In this section, we test the proposed deep CNN and DFS method for cascading outage screening on the IEEE 57-bus system and on the European 1354-bus system. The deep CNN is first implemented as a regression model of ACOPF. Then, the scenario tree and DFS algorithm are deployed for fast cascading outage path screening.

A. Deep CNN regression of ACOPF model

The structure of the proposed deep CNN has been demonstrated in Fig. 2. For scenario uncertainties, we assume that the variation of load forecast error follows a normal distribution with zero mean and a standard deviation of 0.1. In this study, we consider at most three cascading outage stages, i.e. $k = 3$. The number of operation scenarios and possible line outage scenarios considered in generating the training set and the test set are summarized in TABLE I:

TABLE I Summary of Training/Test set Generation

Test case	Training set				Test set			
	No. of scenarios	Stage 1	Stage 2	Stage 3	No. of scenarios	Stage 1	Stage 2	Stage 3
57-bus	33	10	10	10	7	10	10	10
1354-bus	61	20	20	20	14	20	20	20

TABLE I is explained as follows: taking the IEEE 57-bus system as an example, for the training set, we have 33 different load scenarios at the initial stage. At each outage stage, 10 possible line outage selections are considered based on their failure probability. As such, the total number of training samples will be $N_s + N_s \times N + N_s \times N^2 + \ldots\ldots + N_s \times N^k = N_s \times (N^{(k+1)} - 1)/(N - 1) = 33\times(10^4-1)/(10-1) = 36{,}663$, where N_s is the number of load scenarios, in this case 33; and N is the number of possible line outages, in this case 10. However, under some circumstances when the ACOPF does not converge, such samples are removed from the above training sets. The same explanation applies for other figures in the table.

Note that part of the training set is used as the validation set. For both systems, 20% of the training samples are used as the validation set. The difference between the validation set and the test set is that the validation set has load scenarios that are also included in the training set, while the test set has different load scenarios from those in the training set (but follows the same probability distribution). The deep CNN's accuracy is verified by both sets to prove its generalization under different instances.

All the samples are generated by the MATLAB toolbox MATPOWER [37]. The hardware environment is an Nvidia GeForce GTX 1080 Ti Graphic Card with 11 GB memory and 1.582 GHz core clock. The software environment is the open-source deep learning platform TensorFlow. The learning rate is set to 1e-3, and the number of training epochs is set to 500. To improve the deep CNN regression accuracy, a repeated training process is conducted. Taking the 57-bus system as an example, the training process for the deep CNN is repeated three times, and each time the learning rate is scaled down by 10 times of its previous value. This means that the deep CNN is first trained for 500 epochs with the learning rate 1e-3, and the trained model is saved. Then the deep CNN is trained for another 500 epochs with the saved model as the initial value and a learning rate of 1e-4, and then the new trained model is saved. The above process repeats three times. For the 1354-bus system, the process repeats four times. With repeated training, the algorithm can fine search within the local area with a smaller learning rate to avoid overshooting. The final training results and the test results are shown in TABLE II-TABLE III:

TABLE II Sample Set Size for Deep CNN Training and Testing

Case	Training set size	Validation set size	Test set size
57-bus	24,620	6,155	5,497
1354-bus	18,680	4,670	5,278

TABLE III ACOPF Regression Results based on Deep CNN

Case	Validation set error			Test set error		
	v	θ	SI(%)	v	θ	SI(%)
57-bus	5.3e-4	9.5e-4	0.65	6.2e-4	1.8e-3	2.80
1354-bus	2.8e-4	2.6e-4	0.09	1.6e-4	2.4e-4	0.15

In TABLE III, the error of v and θ is the mean absolute difference between the actual value and the estimated value produced by the deep CNN, and the error of the security index is the mean relative percentage error. As shown in the table, the error measurement is considerably smaller for both systems, which demonstrates the accuracy of deep CNN regression.

To illustrate the high computational efficiency of the deep CNN, we compare ACOPF runtime of the 5,497 and 5,278 test samples between deep CNN regression and the model-based method in MATPOWER, and the results are summarized in TABLE IV:

TABLE IV Test Time Comparison

Case	Training time(s)	Test time (s) (deep CNN)	Test time (s) (model-based)	Acceleration ratio
57-bus	906	0.16	225.85	**1,412**
1354-bus	24,692	2.73	8,185	**2,998**

TABLE IV shows that the computation speed of the deep CNN is thousands of times faster than that of the traditional model-based ACOPF. This is because once the deep CNN is well-trained, it has formulated high dimensional mapping between input and output, and it can directly generate optimal power flow results for new instances with different loading conditions and system topology changes, without incurring the iterative calculation. This computation-free feature makes the deep CNN an advantageous tool for solving highly complex large-scale power system planning and operation problems, where the model-based method can be excessively time- and resource-consuming. In addition, the training time for both test cases are within an acceptable range, given that the training for the deep CNN is completed off-line.

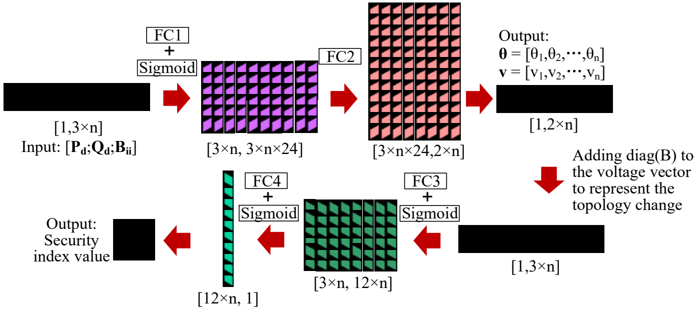


Fig. 6. ANN structure for security assessment

To further validate the high learning ability of the proposed deep CNN, we designed a traditional artificial neural network (ANN) with fully-connected layers as a comparison for cascading outage screening. The configuration of the proposed ANN is shown in Fig. 6.

The difference between the proposed deep CNN model and the traditional ANN model is that the former utilizes convolutional layers to extract features, while the latter utilizes the fully connected layers. In addition, the deep CNN has multiple hidden layers for sufficient feature extraction, while in the ANN there is only one hidden layer between the input and the output, e.g., FC1 and FC3 are the hidden layers in Fig. 6. The same training set, validation set, and test set are used for ANN training and testing. The ANN is also trained repeatedly for the same number of epochs for fair comparison. The final regression results of the ANN are shown in TABLE V:

TABLE V ACOPF REGRESSION BASED ON TRADITIONAL ANN

Case	Validation set error			Test set error		
	v	***θ***	**SI(%)**	***v***	***θ***	**SI(%)**
57-bus	8.3e-4	1.9e-3	4.62	1.0e-3	3.0e-3	8.78
1354-bus (Eu.)	-	-	-	-	-	-

The results of the 1354-bus system are not available for ANN because the large-scale system causes the size of the FC layer parameters to exceed the memory limit. For the 57-bus system, the deep CNN provides more accurate results than the traditional shallow ANN. This is because the convolutional kernels within the deep CNN utilize sparse connectivity to extract better features for model regression. In addition, the number of parameters in the convolutional layers is much lower than that of the FC layers, which spares both the computation source and the storage source.

B. Identifying cascading outage path with DFS algorithm

The function of the deep CNN is to evaluate the system security status for each operation scenario during cascading outages. In this subsection, a scenario tree is first constructed to represent multiple realizations of real-time uncertainties. Then the security index of each node in the scenario tree is calculated based on the estimated results from the deep CNN. Finally, the DFS algorithm is applied to evaluate the severity of each cascading outage path along the entire scenario tree.

Two scenario trees for the IEEE-57 bus system and European 1354-bus system are constructed based on their respective test sets. For the 57-bus system, because no line capacity data is given, the alarm limit is set at 1.35 times the line flow under normal conditions, and the security limit is set at 1.4 times the line capacity, which follows [30]; for the 1354-bus system, the alarm limit is set at 1.35 times the original line capacity, and the security limit is set at 1.4 times the line capacity. The results of the cascading outage screening are shown in TABLE VI-TABLE VII.

TABLE VI TIME EFFICIENCY OF DFS ALGORITHM

Case	No. of paths	Time(s)	Average $\mathrm{SI}^{(1)}_{\exp}$ error (%)
57-bus	4,856	0.019	1.06
1354-bus	4,424	0.010	0.16

In TABLE VI, the fourth column is the average relative error of the accumulative security index $\mathrm{SI}^{(1)}_{\exp}$ based on the estimated results from the deep CNN compared with the actual ACOPF results for all the cascading outage paths. It can be seen that the average error rates for the two test cases are considerably small, which further indicates that the deep CNN regression results can be utilized as a reliable index for cascading outage severity evaluation.

The DFS algorithm is written in MATLAB R2017b, and the hardware environment is an Nvidia GeForce GTX 1080 Ti Graphic Card with 11 GB memory and 1.582 GHz core clock. As seen in the third column of TABLE VI, the calculation time of $\mathrm{SI}^{(1)}_{\exp}$ for all the cascading outage paths in both test cases takes no more than 0.02 second, which demonstrates the high computational efficiency of the DFS algorithm.

TABLE VII presents the cascading outage path with the highest $\mathrm{SI}^{(1)}_{\exp}$ in the two test cases, which indicates their highest severity. "Actual" means the result is based on the real security index value for calculating $\mathrm{SI}^{(1)}_{\exp}$, and "Estimated" means that the result is based on the value from the deep CNN for calculating $\mathrm{SI}^{(1)}_{\exp}$. In the third column, e_{load} stands for the load forecast error. For example, in the 57-bus system, the severest cascading outage path is the 7th scenario with a load forecast error of 0.0725, with line 9-11 tripped at the 1st outage stage, line 9-13 tripped at the 2nd outage stage, and line 3-15 tripped at the 3rd outage stage. As can be observed from the table, in the 57-bus system, the actual cascading outage path is the same as the estimated cascading outage path; in the 1354-bus system, the estimated path is different from the actual path in the third outage stage. However, it is found that the estimated path has the third highest $\mathrm{SI}^{(1)}_{\exp}$ if using the actual security index for calculating, which is only 0.0025% smaller than the highest $\mathrm{SI}^{(1)}_{\exp}$. Therefore, it can be safely concluded that the computation-free deep CNN is accurate enough to serve as a highly efficient tool for fast cascading outage screening in combination with the DFS algorithm, especially for large-scale power systems with multiple uncertain scenarios.

TABLE VII CASCADING OUTAGE SCREENING RESULTS

Case		Scenario	Stage 1	Stage 2	Stage 3
57-bus	Actual	7 (e_{load}: 0.0725)	L 9-11	L 9-13	L 3-15
	Estimated	7 (e_{load}: 0.0725)	L 9-11	L 9-13	L 3-15
1354-bus	Actual	13 (e_{load}: -0.0865)	L 2426-8961	L 1146-7945	L 6806-1609
	Estimated	13 (e_{load}: -0.0865)	L 2426-8961	L 1146-7945	L 3248-7309

Some further insights can be gained from the cascading outage screening results. In TABLE VIII, we analyze the transmission lines that are most frequently tripped at each cascading outage stage in the first 100 cascading outage paths with the highest estimated $\mathrm{SI}^{(1)}_{\exp}$ value, and also compare with the results based on actual $\mathrm{SI}^{(1)}_{\exp}$ value. The line index marked in red bold font represents the lines that are missed in the estimated line set. As shown in the table, almost all the lines except one in the actual line set are detected in the estimated line set, which again proves the accuracy of deep CNN regression. The information revealed in the table can be used as a reference for system operators to take predictive measures for line capacity expansion or load shedding in advance to improve system operation security against cascading risks.

TABLE VIII INDEX OF MOST FREQUENTLY TRIPPED LINES IN EACH CASCADING OUTAGE STAGE

Case 57		
Stage 1	Actual lines	L 9-12, L 12-13, L 11-13, L 9-13, L 4-6, L 9-11
	Estimated lines	L 9-12, L 12-13, L 11-13, L 9-13, L 4-6, L 9-11
Stage 2	Actual lines	L 5-6, L 3-15, L 9-10, L 12-13, L 9-12, L 11-13, L 4-6, L 9-11, L 9-13
	Estimated lines	**L 5-6**, L 3-15, L 9-10, L 12-13, L 9-12, L 11-13, L 4-6, L 9-11, L 9-13
Stage 3	Actual lines	L 24-26, L 26-27, L 19-20, L 9-11, L 12-13, L 11-13, L 48-49, L 13-14, L 23-24, L 4-6, L 3-15, L 9-13, L 9-10, L 5-6, L 9-12

	Estimated lines	L 24-26, L 26-27, L 19-20, L 9-11, L 12-13, L 11-13, L 48-49, L 13-14, L 23-24, L 4-6, L 3-15, L 9-13, L 9-10, L 5-6, L 9-12
Case 1354		
Stage 1	Actual lines	L 3248-7309, L 4689-4936, L 6629-7309, L 1146-7945, L 2426-8961
	Estimated lines	L 3248-7309, L 4689-4936, L 6629-7309, L 1146-7945, L 2426-8961
Stage 2	Actual lines	L 2426-6888, L 6629-7309, L 3248 –7309, L 4689-4936, L 1146-7945, L 2426-8961
	Estimated lines	L 2426-6888, L 6629-7309, L 3248 –7309, L 4689-4936, L 1146-7945, L 2426-8961
Stage 3	Actual lines	L 2426-6888, L 1146-7945, L 6629-7309, L 3248-7309, L 4689-4936, L 2426-8961, L 6806-1609
	Estimated lines	L 2426-6888, L 1146-7945, L 6629-7309, L 3248-7309, L 4689-4936, L 2426-8961, L 6806-1609

VI. Conclusions

In this paper, a data-driven fast cascading outage screening approach is proposed based on deep convolutional neural network (deep CNN) and depth-first search (DFS). The deep CNN is constructed as a regression tool to estimate the ACOPF results under different contingencies and also the system security index. The DFS algorithm is applied to scan the scenario tree to detect the severest cascading outage path based on the estimated security index value provided by deep CNN. Simulation results on the IEEE 57-bus and European 1354-bus systems verify the high accuracy and high computational efficiency of the proposed method. The practical implications of the study are summarized as follows:

1) As the penetration of uncertainties into the bulk power system increases, the number of operation scenarios to be examined for system security assessment will grow exponentially, resulting in an unbearable computational cost to conventional model-based methods. The proposed, nearly computation-free data-driven method can quickly detect system vulnerability under multiple scenarios. The high accuracy and computational efficiency make the proposed method a desirable choice for real-time system screening.

2) With historical cascading outage data provided as a training set, the proposed data-driven method can easily adapt to power systems with different scales and multiple outage stages. The flexibility and scalability give the proposed method the potential to be developed as a general cascading outage screening tool in real-world applications.

3) The screening results of the deep CNN and the DFS method can serve as a reference for power system operators to take preventive measures against latent outages, and to reduce the system risk management costs such as load shedding and generator redispatch. The screening results can also be used as guidelines for future power system planning to efficiently allocate investments to the most vulnerable transmission devices.

References

[1] Glossary of Terms Used in NERC Reliability Standards. [Online]. Available: https://www.nerc.com/files/glossary_of_terms.pdf

[2] WSCC Operations Committee, "Western Systems Coordinating Council Disturbance Report For the Power System Outages that Occurred on the Western Interconnection on July 2, 1996 and July 3, 1996," Western Syst. Coordinating Council, Salt Lake City, UT, USA, Tech. Rep., 1996.

[3] U.S.-Canada Power System Outage Task Force, "Final report on the August 14th blackout in the United States and Canada," Apr. 2004.

[4] Protection System Response to Power Swings, System Protection and Control Subcommittee, NERC, Atlanta, GA, USA, Aug. 2013. [Online]. Available: http://www.nerc.com

[5] Transmission System Planning Performance Requirements, NERC Standard TPL-001-4, 2013.

[6] I. Dobson, B.A. Carreras, V.E. Lynch, and D.E. Newman,"Complex systems analysis of series of blackouts: cascading failure, critical points, and self-organization," *Chaos*, vol. 17, 026103, June 2007.

[7] S. Mei, Y, Ni. Weng, G. Wang, and S. Wu, "A study of self-organized criticality of power system under cascading failures based on AC-OPA with voltage stability margin," *IEEE Trans. Power Syst*., vol. 23, no. 4, pp. 1719–1726, Nov. 2008.

[8] D. S. Kirschen, D. Jawayeera, D. P. Nedic, and R. N. Allan, "A probabilistic indicator of system stress," *IEEE Trans. Power Syst.*, vol. 19, no. 3, pp. 1650–1657, Aug. 2004.

[9] I. Dobson, B. A. Carreras, and D. E. Newman, "A loading dependent model of probabilistic cascading failure," Probability Eng. Informational Sci., vol. 19, no. 1, pp. 15–32, Jan. 2005.

[10] P. Henneaux, P.-E. Labeau, J.-C. Maun, and L. Haarla, "A Two-Level Probabilistic Risk Assessment of Cascading Outages," *IEEE Trans. Power Syst*, vol. 31, no. 2, pp. 2393-2403, 2016.

[11] R. Yao, S. Huang, K. Sun, F. Liu, X. Zhang, and S. Mei, "A Multi-Timescale Quasi-Dynamic Model for Simulation of Cascading Outages," *IEEE Trans. Power Syst*., vol. 31, no. 4, pp. 3189-3201, 2016.

[12] J. Guo, F. Liu, J. Wang, J. Lin, and S. Mei, "Toward Efficient Cascading Outage Simulation and Probability Analysis in Power Systems," *IEEE Trans. Power Syst.*, vol. 33, no. 3, pp. 2370-2382, 2018.

[13] M. Rahnamay-Naeini, and M. M. Hayat, "Cascading Failures in Interdependent Infrastructures: An Interdependent Markov-Chain Approach," *IEEE Trans. Smart Grid*, vol. 7, no. 4, pp. 1997-2006, 2016.

[14] P. Hines, I. Dobson, and P. Rezaei, "Cascading Power Outages Propagate Locally in an Influence Graph that is not the Actual Grid Topology," *IEEE Trans. Power Syst.*, vol. 32, no. 2, pp. 958-967, Mar. 2017.

[15] J. Qi, W. Ju, and K. Sun, "Estimating the Propagation of Interdependent Cascading Outages with Multi-Type Branching Processes," *IEEE Trans. Power Syst*., vol. 32, no. 2, pp. 1212-1223, Mar. 2016.

[16] J. Qi, J. Wang, and K. Sun, "Efficient Estimation of Component Interactions for Cascading Failure Analysis by EM Algorithm," *IEEE Trans. Power Syst*., vol. 33, no. 3, pp. 3153-3161, 2018.

[17] R. Yao, S. Huang, K. Sun, F. Liu, X. Zhang, S. Mei, W. Wei, and L. Ding, "Risk Assessment of Multi-Timescale Cascading Outages Based on Markovian Tree Search," *IEEE Trans. Power Syst.*, vol. 32, no. 4, pp. 2887-2900, Jul. 2017.

[18] R. Yao, and K. Sun, "Toward Simulation and Risk Assessment of Weather-Related Outages," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 4391-4400, 2019.

[19] J. Guo, F. Liu, J. Wang, M. Cao, and S. Mei, "Quantifying the Influence of Component Failure Probability on Cascading Blackout Risk," *IEEE Trans. Power Syst.*, vol. 33, no. 5, pp. 5671-5681, 2018.

[20] X. Liu, and Z. Li, "Revealing the Impact of Multiple Solutions in DCOPF on the Risk Assessment of Line Cascading Failure in OPA Model," *IEEE Trans Power Syst.*, vol. 31, no. 5, pp. 4159-4160, Sep. 2016.

[21] Z. Ma, C. Shen, F. Liu, and S. Mei, "Fast Screening of Vulnerable Transmission Lines in Power Grids: A PageRank-Based Approach," *IEEE Trans. Smart Grid*, vol. 10, no. 2, pp. 1982-1991, 2019.

[22] X. Wei, J. Zhao, T. Huang, and E. Bompard, "A Novel Cascading Faults Graph Based Transmission Network Vulnerability Assessment Method," *IEEE Trans. Power Syst.*, vol. 33, no. 3, pp. 2995-3000, May 2018.

[23] E. J. Anderson, and J. Linderoth, "High Throughput Computing for Massive Scenario Analysis and Optimization to Minimize Cascading Blackout Risk," *IEEE Trans. Smart Grid*, vol. 8, no. 3, pp. 1427-1435, 2017.

[24] A. A. Babalola, R. Belkacemi, and S. Zarrabian, "Real-Time Cascading Failures Prevention for Multiple Contingencies in Smart Grids Through a Multi-Agent System," *IEEE Trans. Smart Grid*, vol. 9, no. 1, pp. 373-385, 2018.

[25] R. Yao, K. Sun, F. Liu, and S. Mei, "Management of Cascading Outage Risk Based on Risk Gradient and Markovian Tree Search," *IEEE Trans. Power Syst.*, vol. 33, no. 4, pp. 4050-4060, 2018.

[26] R. Sunitha, S. K. Kumar, and A. T. Mathew, "Online static security assessment module using artificial neural networks," *IEEE Trans. Power Syst.*, vol. 28, no. 4, pp. 4328-4335, Nov. 2013.

[27] A. Gupta, G. Gurrala, and S. P.S, "An Online Power System Stability Monitoring System Using Convolutional Neural Networks," *IEEE Trans. Power Syst.*, vol. 34, no. 2, pp. 864-872, Mar. 2019.

[28] M. Sun, I. Konstantelos, and G. Strbac, "A Deep Learning-Based Feature Extraction Framework for System Security Assessment," *IEEE Trans. Smart Grid*, vol. 10, no. 5, pp. 5007-5020, Sep. 2019.

[29] M. R. Salimian, and M. R. Aghamohammadi, "A Three Stages Decision Tree-Based Intelligent Blackout Predictor for Power Systems Using Brittleness Indices," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 5123-5131, 2018.

[30] Y. Jia, Z. Xu, L. L. Lai, and K. P. Wong, "Risk-Based Power System Security Analysis Considering Cascading Outages," *IEEE Trans. Industrial Informatics*, vol. 12, no. 2, pp. 872-882, Apr. 2016.

[31] K. Nara, K. Tanaka, and H. Kodama, etc. "On-line contingency selection algorithm for voltage security analysis," *IEEE Trans. Power Apparatus and Systems*, vol. PAS-104, no. 4, pp. 846-856, Apr. 1985.

[32] R. Sunitha, R. Sreerama Kumar, and A. T. Mathew, "A composite security index for on-line static security evaluation," Elect. Power Compon. Syst., vol. 39, no. 1, pp. 1–14, Jan. 2011.

[33] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Learning," Cambridge: MIT press, 2016, pp. 159-279.

[34] Y. Du, F. Li, J. Li, and T. Zheng, "Achieving 100x Acceleration for N-1 Contingency Screening with Uncertain Scenarios using Deep Convolutional Neural Network," *IEEE Trans. Power Syst.*, vol. 34, no. 4, pp. 3303-3305, Jul. 2019

[35] Y. Du, F. Li, and C. Huang, "Applying Deep Convolutional Neural Network for Fast Security Assessment with N-1 Contingency," in 2019 *IEEE PES General Meeting*, pp. 1-5.

[36] F. Li and Y. Du, "From AlphaGo to Power System AI: What Engineers Can Learn from Solving the Most Complex Board Game," *IEEE Power & Energy Magazine*, vol. 16, no. 2, pp. 76-84, Mar. 2018.

[37] 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 Trans. Power Syst.*, vol. 26, no. 1, pp. 12-19, Feb. 2011.

Yan Du (S'16) received the B.S. degree from Tianjin University, Tianjin, China, and the M.S. degree from Institute of Electrical Engineering, Chinese Academy of Sciences, Beijing, China, in 2013, and 2016, respectively. She is currently working toward the Ph.D. degree at the University of Tennessee, Knoxville, TN, USA. Her research interests include distribution system operation, and deep learning in power systems.

Fangxing Li (S'98–M'01–SM'05-F'17) is also known as Fran Li. He received the B.S.E.E. and M.S.E.E. degrees from Southeast University, Nanjing, China, in 1994 and 1997, respectively, and the Ph.D. degree from Virginia Tech, Blacksburg, VA, USA, in 2001.

Currently, he is the James McConnell Professor at the University of Tennessee, Knoxville, TN, USA. His research interests include deep learning in power systems, renewable energy integration, demand response, and power market and power system computing. Since 2020, he has been the Editor-In-Chief of *IEEE Open Access Journal of Power and Energy*.

Tongxin Zheng (SM'08) received the B.S. degree in electrical engineering from North China University of Electric Power, Baoding, China, in 1993, the M.S. degree in electrical engineering from Tsinghua University, Beijing, China, in 1996, and the Ph.D. degree in electrical engineering from Clemson University, Clemson, SC, USA, in 1999. Currently, he is technical director at the ISO New England, Holyoke, MA, USA. His main interests are power system optimization and electricity market design.

Jiang Li (S'01-M'07-SM'12) received the B.S. degree from Shanghai Jiaotong University, Shanghai, China, the M.S. degree from Tsinghua University, Beijing, China, in 1992 and 2000, respectively, and the Ph.D. degree from University of Texas at Arlington, Arlington, TX, USA, in 2004.

Currently, he is an Associate Professor at Old Dominion University, Norfolk, VA, USA. His research interests include deep learning, computer-aided medical diagnosis systems, remote sensing image analysis, and modeling and simulation.