



A Fully Data-Driven Method Based on Generative Adversarial Networks for Power System Dynamic Security Assessment With Missing Data

Chao Ren , *Student Member, IEEE*, and Yan Xu , *Senior Member, IEEE*

Abstract—This paper proposes a fully data-driven approach for PMU-based pre-fault dynamic security assessment (DSA) with incomplete data measurements. The generative adversarial network (GAN), which is an emerging unsupervised deep learning technique based on two contesting deep neural networks, is used to address the missing data. While the state-of-the-art methods for missing data are dependent on PMU observability, they are limited by the placement of PMU and network topologies. Distinguished from existing methods, the proposed approach is fully data-driven and can fill up incomplete PMU data independent on PMU observability and network topologies. Therefore, it is more generalized and extensible. Simulation results show that, under any PMU missing conditions, the proposed method can maintain a competitively high DSA accuracy with a much less computation complexity.

Index Terms—Data-driven, dynamic security assessment, generative adversarial networks, hybrid ensemble learning.

I. INTRODUCTION

A. Background and Motivation

INTELLIGENT System (IS) techniques have been identified as an effective approach to make use of wide-area measurements to achieve data-driven DSA. The principle is to train a learning model (e.g., a classifier or predictor) from a comprehensive DSA database. Once well trained, it can be applied for on-line pre-fault DSA with real-time power system measurements, providing advantages of much faster assessment speed, less data requirement, stronger generalization capability [1].

With the increasing integration of renewable energy sources and active demand-side response, more uncertainties are introduced to power system, which significantly challenged the power system's dynamic security assessment. Traditionally, it is assumed that the DSA model input (e.g., PMU measurements)

is complete for training and always online available for application [2], [3]. However, this assumption may not hold due to many practical issues, such as PMU malfunction, communication congestion/failure, or even cyber-attack, etc [4]–[8]. While traditional methods are all based on the complete data input, they will become ineffective if the data is missing. Thus, there is a pressing need for *missing-data tolerant* methods to maintain a satisfactory performance under the missing data conditions.

B. Literature Review

In the literature, decision tree with surrogate (DTWS) [4], degenerate decision tree [5], feature ensemble learning [6], and feature estimation [7] have been proposed to alleviate the detrimental effect of missing data. The DTWS [4] uses the neighboring tree node to replace the missing data, but it usually suffers from an unsatisfactory performance due to the loss information. In [5], the degenerate decision tree method employs a large number of single decision trees, whose performance is sensitive to the tree structure. By assembling small DTs which are trained by location-separated random subspace, it achieves better DSA viability compared to single DT methods. In [7], the feature estimation method can predict the missing data directly, but it needs to train a large number of classifiers for different PMU missing conditions which would suffer from curse of dimensionality. In [6], [8], a robust feature ensemble model is designed to strategically collect observation-constrained PMU clusters as training database. Such database is further utilized to train single DSA classifiers. Under any PMUs missing conditions, the ensemble of available classifiers could achieve maximum system observability with least single classifiers, thus could maintain a high accuracy and computational efficiency.

However, it is clear that these methods fully depend on the PMU observability and network topologies, once they are changed or the IS cannot expect the detailed placement of missing data in advance, the model may become ineffective and has to be updated.

C. Contributions in This Paper

To address the missing data problem, this paper proposes a *fully data-driven* method based on an emerging deep-learning technique, called generative adversarial network (GAN) [9]–[11]. The principle is to develop a GAN model to directly and accurately fill up the missing data without depending on

Manuscript received January 8, 2019; revised April 10, 2019; accepted June 9, 2019. Date of publication June 13, 2019; date of current version October 24, 2019. This work was supported in part by the Ministry of Education, Republic of Singapore, under Grant AcRF TIER 1 2017-T1-001-228 (RG92/17), and in part by the National Research Foundation of Singapore under Project NRF2018-SR2001-018. The work of Y. Xu was supported by the Nanyang Assistant Professorship from the Nanyang Technological University, Singapore. Paper no. TPWRS-00032-2019. (Corresponding author: Yan Xu.)

C. Ren is with the Interdisciplinary Graduate School, Nanyang Technological University, Singapore 639798 (e-mail: RENC0003@e.ntu.edu.sg).

Y. Xu is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: eeyanxu@gmail.com).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TPWRS.2019.2922671

TABLE I
NOMENCLATURE

Notation	Description	Notation	Description
G	Generator	θ_g	Generator weights
D	Discriminator	θ_d	Discriminator weights
$V(\cdot)$	Value function	x	True Dataset
p_g	True sample distribution	z	Input vector
x	Single sample	γ	GAN learning rate
m	Number of ELM or RVFL	φ	Clipping parameter
n_{discr}	Number of discriminator iterations	w	Weight
n_{gene}	Number of generator iterations	b	Biases
ρ_1, ρ_2	Exponential decay rates	ϵ	Prevent parameters

PMU observability and network topologies. Thus, the method is more generalized and extensible. Besides, any machine learning algorithm can be used for the DSA model. Specifically, the proposed IS combines a GAN model and a DSA model. By constantly updating and fixing GAN model, it can collectively provide a complete data set against missing data. The DSA model can be a classifier or predictor using the any effective machine learning algorithm in an ensemble form, in this paper as extreme learning machine (ELM) and random vector functional link networks (RVFL), to obtain the more diversified machine learning outcome.

Moreover, a high-performance stochastic optimization gradient descent algorithm *Adam* [12] is applied in this paper for weights updates during the GAN model training process, to make the training process more accurate and faster. The main contributions of the proposed method are as follows:

- 1) A GAN model is developed to generate the missing data without depending on PMU observability and network topologies, and *Adam* algorithm is used to pursue the highest assessment accuracy and efficiency.
- 2) A hybridized random learning model is developed for DSA. Multiple randomized learning algorithms are ensemble to improve the learning diversity. Their aggregated output tends to be more accurate and more robust than the performance of using a single learning algorithm.

The proposed method has been tested on New England 10-machine 39-bus system and demonstrated higher accuracy compared to other state-of-art methods. By offering the system operators more flexibility in manipulating the assessment performance, the proposed IS can accommodate different PMU missing conditions.

II. PROPOSED METHODOLOGY

The proposed data-driven IS is based on one unsupervised learning algorithm (GAN) and two randomized learning algorithms (ELM and RVFL). This section firstly describes the fundamentals of these three different learning algorithms, then introduce the whole IS and *Adam* optimization algorithm. The nomenclature is given in Table I.

A. Generative Adversarial Network

As an emerging unsupervised machine learning algorithm, GAN can directly generate realistic features based only on the

training data set without the need to fit an existing explicit model [9]. GAN is implemented with two deep neural networks, called *generator* and *discriminator*, which contest with each other in a zero-sum game framework.

Specifically, the generator is to produce data samples that follows the distribution of the historical training data, while the discriminator is to distinguish between the generated data samples and the true historical data. By training the generator and the discriminator iteratively, GAN can achieve an equilibrium that the discriminator can no longer distinguish between the generated and historical data, which means the generated data is aligned with the distribution of the historical data. For the missing PMU data problem in this paper, GAN is to generate data that can accurately replace the missing data.

Generator: Denote the true sample distribution of the observation features as p_g over dataset x , we define a prior for input noise vector z under a given distribution $z \sim p_z(z)$ which can be easily sampled from (e.g., incomplete PMU measurements). The objective is to find a function G following $p_g(x)$ as $G(z; \theta_g)$ after transformation, where G is a differentiable function represented by a multilayer perceptron.

Discriminator: Denote a second multilayer perceptron $D(x; \theta_d)$ that outputs a single scalar, where $D(x)$ represents the probability that x came from the generate data rather than p_g , and thus to maximize the probability of the correct label between $\mathbb{E}[D(x)]$ (real data) and $\mathbb{E}[D(G(z))]$ (generated data).

Here, θ_g and θ_d denote the weights of generator and discriminator neural networks, respectively.

The structure of GAN is illustrated in Fig. 1, and its training process is detailed in Algorithm 1. The GAN training can be divided into two steps: updating discriminator with fixed generator parameter and updating generator with fixed discriminator parameter. As the objectives for generator and discriminator defined above, GAN needs to update neural networks' weights based on the loss function. The loss functions for discriminator and generator are respectively formulated as:

$$\max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log (1 - D(G(z)))] \quad (1)$$

$$\max_G V(D, G) = \mathbb{E}_{z \sim p_z(z)} [\log (D(G(z)))] \quad (2)$$

where Eq. (2) is equal to Eq. (3) as follows:

$$\min_G V(D, G) = \mathbb{E}_{z \sim p_z(z)} [\log (1 - D(G(z)))] \quad (3)$$

Then we combine Eq. (1) with Eq. (3) to formula a two-player iterative minimax game with value function $V(G, D)$:

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log (1 - D(G(z)))] \quad (4)$$

This objective function can produce the same fixed point of dynamic of generator G and discriminator D , and provides much stronger gradients in early learning process. When G is less, D can reject samples with high confidence owing to their obvious difference from the training data. In this case, $\log(1-D(G(z)))$

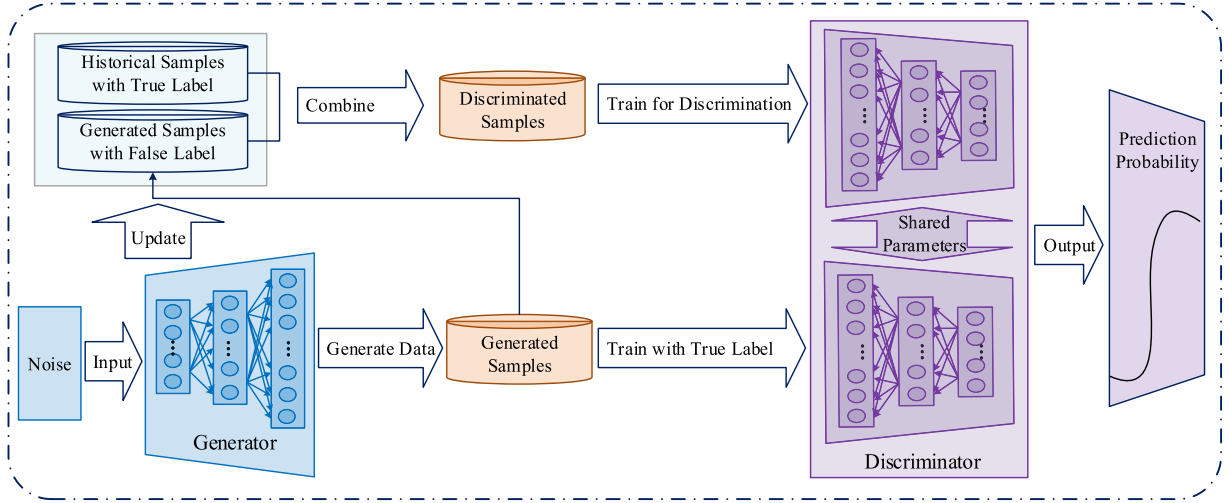


Fig. 1. Structure of the GAN.

Algorithm 1: GAN Training Process.

Input: Learning rate γ , minibatch size m , clipping parameter ϕ , number of iterations for discriminator n_{discr} and generator n_{gener}

Output: Complete data $O_{complete}(x)$ filled by generated data $O_{generator}(x)$

Initialize: Initial generator weight θ_g and discriminator θ_d

for number of training iterations **do**

for $i = 0, \dots, n_{discr}$ **do**

Update discriminator parameter, fix generator parameter

Sample minibatch of m examples from historical data:

$\{x^{(t)}\}_{t=1}^m \sim p_{data}(x)$

Sample minibatch of m noises from noise prior:

$\{z^{(t)}\}_{t=1}^m \sim p_g(z)$

Update discriminator networks by descending its gradient:

$g_{\theta_d} \leftarrow \nabla_{\theta_d} \frac{1}{m} \sum_{t=1}^m [\log D(x^{(t)}) + \log(1 - D(G(z^{(t)})))]$

$\theta_d \leftarrow \theta_d - \gamma \cdot Adam(\theta_d, g_{\theta_d})$

$\theta_d \leftarrow clip(\theta_d, \phi, -\phi)$

end for

for $j = 0, \dots, n_{gener}$ **do**

Update generator parameter, fix discriminator parameter

Sample minibatch of m noises from noise prior:

$\{z^{(t)}\}_{t=1}^m \sim p_g(z)$

Update generator networks by descending its gradient:

$g_{\theta_g} \leftarrow \nabla_{\theta_g} \frac{1}{m} \sum_{t=1}^m [\log(1 - D(G(z^{(t)})))]$

$\theta_g \leftarrow \theta_g - \gamma \cdot Adam(\theta_g, g_{\theta_g})$

end for

end for

The gradient-based update can use any standard gradient-based learning rule. In this model, *Adam* algorithm is applied here.

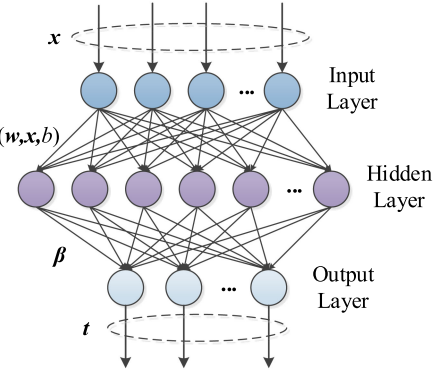


Fig. 2. The structure of ELM.

F2. The global minimum of $\max_D V(G, D)$ is achieved if only if:

$$p_g(x) = p_{data}(x) \quad (6)$$

The proof is provided in [9].

As mentioned before, $D(x; \theta_d)$ and $G(z; \theta_g)$ are completely different functions, thus any standard gradient-based learning rule for these two networks can be applied to optimize their performances.

B. Neural Networks With Random Weights and Biases

Randomized learning neural networks were widely employed which were naturally diverse and showed the fast learning implementation for single-hidden layer feedforward network (SLFN) [13]. Among them, random vector functional link neural network and extreme learning machine are two more popular neural networks with random weights and biases.

ELM was proposed by Huang [14], and Fig. 2 shows the structure of ELM. It includes three layers: input layer, hidden layer and output layer. For instances with n -dimension input vector and m -dimension target vector, the output function of

saturates. Rather than training G to minimize $\log(1 - D(G(z)))$ we can train G to maximize $\log D(G(z))$.

We consider the optimal discriminator D for any given generator G . The value function $V(G, D)$ can reach the global maximum optimization when satisfying two conditions *F1* and *F2* as follows:

F1. For G fixed, the optimal discriminator D is:

$$D_G^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)} \quad (5)$$

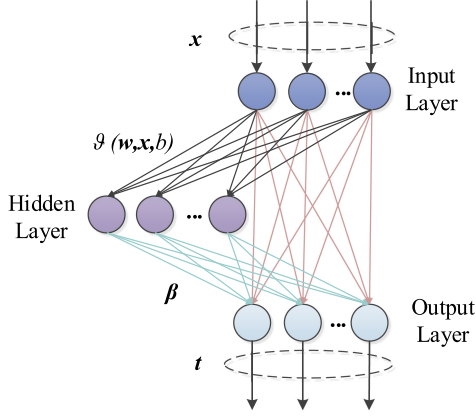


Fig. 3. The structure of RVFL.

ELM with N hidden nodes and activation function g can be mathematically modelled as follows:

$$f_{\tilde{N}}(x_j) = \sum_{i=1}^{\tilde{N}} \beta_i \cdot g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i) = t_j, j = 1, 2, \dots, N \quad (7)$$

where the \mathbf{x}_j is the j -th instance input vector out of N total instances, $\mathbf{x} \in R^n$; t_j is the j -th instance target vector, $t \in R^m$; \mathbf{w}_i is the vector of weight linking input nodes to the i -th hidden node; b_i is the threshold of the i -th hidden node, which could also be written in the form of \mathbf{w}_0 ; β_i is the weight vector from i -th hidden nodes to the output nodes.

In ELM theory, \mathbf{w} and \mathbf{b} could be randomly assigned, then the unknown variable vector only remains β . In that way, by minimizing the training error between output and target value, the output of hidden layer could be determined, which dramatically simplify the training process. More specifically, a compact format of Eq. (8) is given as follows:

$$\mathbf{H}\beta = \mathbf{T} \quad (8)$$

where \mathbf{H} , called hidden layer output matrix, is determined by input weights and input vectors.

RVFL was proposed by Pao [15] and has been widely applied in various fields. The basic structure of RVFL is demonstrated in Fig. 3. The distinctive feature of RVFL is its direct input-output connection. Similar to the format of ELM, the output weights could be determined by solving the following equation:

$$\mathbf{t}_i = \mathbf{h}_i^T \beta, \quad i = 1, 2 \dots N \quad (9)$$

where \mathbf{t} represents the instance target vector, N is the number of input instances, \mathbf{h} represents the vector version of the concatenation of the input layer nodes and the random features in hidden layer, β represents the output weight vector. Assuming that each instance has P input features and Q hidden nodes, there are totally $(P+Q)$ nodes connected to output layer, thus β is comprised of $(P+Q)$ output weights.

The input weights and bias of ELM and RVFL are randomly generated in a suitable domain. Compared with the traditional iterative learning algorithms, ELM and RVFL both show much faster learning speed and requires much less computation

Algorithm 2: Learning Unit Training.

Input: Number of input for instances S and features F , $2m$ classifiers (including m ELMs and m RVFLs).

Output: Constitute the hybrid randomized ensemble model.

Initialize: Initial effective activation function.

Randomly choose instances $s \in [1, S]$ from the database.

Randomly choose features $f \in [1, F]$ from the feature set.

for $i = 1$ to m **do**:

Randomly assign ELM hidden layer nodes h_E within specific optimal range $[h_{MIN}, h_{MAX}]$ (Subject to a tuning process).

Randomly assign RVFL hidden layer nodes h_R within specific optimal range $[h_{MIN}, h_{MAX}]$ (Subject to a pre-tuning process).

Train the ELM and RVFL with the chosen instances, features, the quantity of hidden layer nodes, and activation function.

end for

Combine m ELMs outputs with m RVFLs outputs.

memory, whether categorical classification or numeric prediction. Besides, ELM and RVFL are excellent generalization ability and less need for parameter tuning, such as learning rate setting and stopping criteria design. The core difference between these two different neural networks is that RVFL has a direct input-output link on top of the input-hidden-output structure of ELM. Under such structure, RVFL can map both linear and nonlinear relationships between input and output nodes, which helps regularize the error incurred by the randomness in the input weights and biases and thereby statistically improve the accuracy [16], [17].

C. Hybrid Ensemble Learning

Ensemble learning is to combine multiple learning units [18], [19]. Under such paradigm, the entire model is more diverse, and the aggregated output from single units tends to be more accurate and more robust.

In this paper, a hybrid ensemble learning model is designed for DSA. It combines two different randomized learning algorithms (ELM and RVFL) as the single leaning unit in order to construct the hybrid randomized ensemble model for achieving higher performance, which can increase the accuracy to allow for more flexible and reliable mechanism before failure [20].

The training process for each single learning unit of the DSA ensemble model is showed as Algorithm 2.

D. Rule for Classification

Based on above cases, supposing the hybrid ensemble learning model includes m ELMs and m RVFLs, the prediction outputs from the learning units are combining so as to improve classification reliability. The classification rule of learning unit is showed as Algorithm 3. Y represents the final result and is classified into secure and insecure according to the computational value.

E. Adam Optimization Algorithm

The Adam optimization algorithm is an extension to stochastic gradient descent that computes adaptive learning rates for

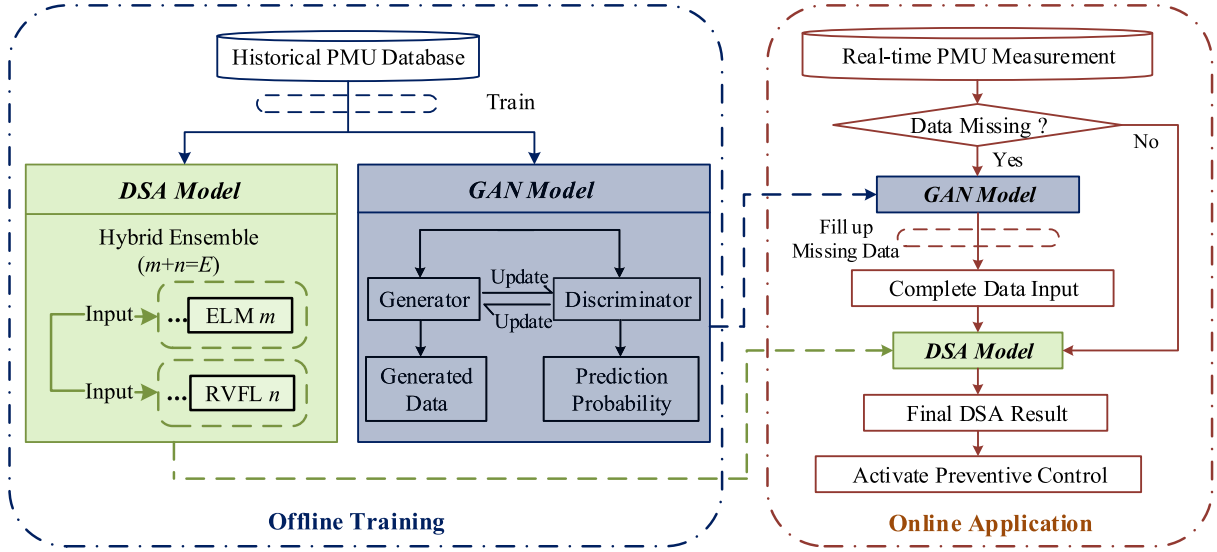


Fig. 4. Framework of the proposed IS.

Algorithm 3: Rule for Classification.**Input:** $2m$ prediction outputs trained by m ELMs and m RVFLs.**Output:** Final result of the hybrid ensemble learning model.

```

for  $i=1$  to  $2m$  do
    if  $y_i > 0$ ,  $y_i = 1 \rightarrow$  output of this single learner is "secure"
    else  $y_i = -1 \rightarrow$  output of this single learner is "insecure"
    end if
end for
if  $Y = \sum_{i=1}^{2m} y_i > 0 \rightarrow$  final decision is "secure"
else  $Y = \sum_{i=1}^{2m} y_i < 0 \rightarrow$  final decision is "insecure"
end if

```

where y_i is the output of the i th single learner, $i=1 \dots 2m$. Y is the final result of the hybrid ensemble learning model.

each parameter [12]. *Adam* optimization algorithm combines the best properties of the *AdaGrad* and *RMSProp* algorithms to handle sparse gradients on noisy problems. In addition to storing an exponentially decaying average of past squared gradients *RMSProp*, *Adam* also keeps counteracting these biases by computing bias-corrected first and second moment estimates. Instead of adapting the parameter learning rates based on the average first moment (the mean) as in *RMSProp*, *Adam* also makes use of the average of the second moments of the gradients (the uncentered variance). The work process is summarized in Algorithm 4.

III. IMPLEMENTATION OF PROPOSED IS

The whole framework of the proposed method can be divided into two stages, including offline training stage and online application stage. The proposed fully data-driven model is illustrated in Fig. 4.

A. Offline Training Stage

At the offline training stage, the DSA database is used to train a GAN model and a DSA model. GAN consists of two deep neural networks, the *generator* and the *discriminator*. By constantly updating and fixing each other, the generator can

Algorithm 4: Adam Optimization Algorithm.**Input:** Learning rate α , exponential decay rates for the momentum ρ_1 and ρ_2 , stochastic objective function $f(\theta)$ with parameter θ , small number to prevent any division by zero ϵ .**Output:** Resulting parameter θ_t .**Initialize:** Initial parameter θ_0 . $m_0 \leftarrow 0$ and $v_0 \leftarrow 0$ # 1st and 2nd moment vector $t \leftarrow 0$ # timestep**while** θ_t not converged **do** $t \leftarrow t + 1$ # Calculate gradient at timestep t $g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$ # Update biased 1st and 2nd moment estimate $m_t \leftarrow \rho_1 \cdot m_{t-1} + (1 - \rho_1) \cdot g_t$ $v_t \leftarrow \rho_2 \cdot v_{t-1} + (1 - \rho_2) \cdot g_t^2$ # Compute bias-corrected 1st and 2nd moment estimate $\hat{m}_t \leftarrow m_t / (1 - \rho_1^t)$ $\hat{v}_t \leftarrow v_t / (1 - \rho_2^t)$ # Update parameter θ_t $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$ **end while**

generate the data which can fill up incomplete PMU data, and the discriminator can calculate the prediction probability of corresponding generated data. After that, they can collectively provide an accurate complete data set against missing data. A high-performance stochastic optimization gradient descent algorithm *Adam* is applied for weights updating in both discriminator and generator neural networks. Clipping parameter is utilized to constrain $D(x; \theta_d)$ to satisfy certain practical conditions as well as preventing gradient explosion. The DSA model is the classifier based on hybrid ensemble learning model of ELM and RVFL.

B. Online Application Stage

At online stage, the real-time PMU measurements are the input features. The IS first determines whether the PMU data is complete. If the data is complete, they will be directly imported

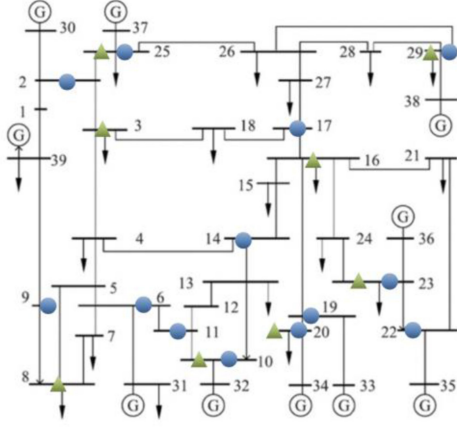


Fig. 5. New England 10-machine 39-bus system with two PMU placement strategies.

TABLE II
PMU PLACEMENT LOCATION

PMU Placement Options	PMU Installation Buses	No. of PMUs
▲ I	3, 8, 10, 16, 20, 23, 25, 29	8
● II	2, 6, 9, 10, 11, 14, 17, 19, 20, 22, 23, 25, 29	13

into the DSA model. Otherwise, the incomplete data will be fed to the GAN, and GAN will fill up the missing data. After that, the filled, complete data will be imported to the DSA model and the classification decision on system's dynamic security status is made by the DSA model. If the system is insecure, preventive control actions such as generation rescheduling should be activated.

IV. NUMERICAL TEST

The proposed method is tested on New England 10-machine 39-bus system as the Fig. 5 [21], where two benchmark PMU placement strategies installed with the minimum number of PMUs required to realize global observability, and corresponding PMU placement locations are listed in Table II [22]. The numerical simulation is conducted on a 64-bit computer with an Intel Core i7 CPU of 2.8-GHz and 16-GB RAM. The proposed algorithms are implemented in the MATLAB platform.

A. Database Generation

Simulations are performed on New England 10-machine 39-bus system to test the proposed method. The operating points are generated using Monte-Carlo method [23]. Specifically, given the forecasted load demand level for each bus, Monte-Carlo simulation is run to sample uncertain power variations. Then, for each uncertain power demand scenario, the power output of the generators is determined by optimal power flow calculation. In doing this, a large number of operating points will be obtained. Then, given the contingency set, time-domain simulation is performed to evaluate the dynamic security label for each operating point. The contingencies considered in the

study are the three-phase faults with inter-area corridor trip and cleared 0.25 s after their occurrences. 10 typical three-phase faults at inter-area corridors are simulated on Transient Stability Analysis Tool (TSAT). Note that any contingencies can also be considered depending on practical needs. Eventually, 5043 operating states with their security conditions are obtained, and the number of secure instances and insecure instances under each contingency are showed in Table III. Thus, there are a total of 50430 instances. In the subsequent work, 80% of the instances were randomly selected as for both DSA model and GAN model training, and the remaining 20% instances form the DSA model testing dataset.

B. Parameter Selection

1) For DSA model:

- Quantity of ELM and RVFL in an Ensemble E : In ensemble learning scheme, the generalization error will decrease as quantity of single learner increase. In this paper, the total number is set as 200 with 100 ELMs and 100 RVFLs.
- Random Parameters in Ensemble Learning: For ELM, activation function is chosen as Sigmoid function and the number of hidden nodes of ELM is better to be randomly selected from [200,400]; For RVFL, the activation function is Sigmoid function, the optimal hidden nodes range is [100,300]. This could also verify that the direct link of RVFL could lead to a thinner model. Sigmoid function is chosen as valid activation hidden optimal range of ELM and RVFL classifiers. And the common optimal hidden node range for ELM and RVFL algorithms is [200, 300].
- Quantity of Training Instances and Testing Instances: The quantity of instances which are selected to train ELMs and RVFLs determines the whole performance of the robustness. In this case study, training instance and testing instance are chosen to be 4034 and 1009, respectively.

2) For GAN model:

- Learning Rate α : Learning rate is a hyper-parameter that controls how much we are adjusting the weights of our network with respect the loss gradient. The value of the learning rate α is decided to be 0.001.
- Exponential Decay Rates for the Momentum ρ_1 and ρ_2 : In this case study, exponential decay rates for the momentum ρ_1 and ρ_2 are set as 0.9 and 0.999, respectively.
- Number to Prevent any Division by zero ϵ : In this case study, ϵ is very small so as to prevent any division and set as 10^{-8} .
- Quantity of Training Instances: In this case study, training instance for GAN model is same as DSA model and is chosen to be 4034.

C. Testing Results

The proposed method is compared with three state-of-the-arts in the literature: robust RVFL ensemble learning [6], DTWS [4],

TABLE III
CONTINGENCY SET

Contingency ID	1	2	3	4	5	6	7	8	9	10
Fault Setting	Fault bus 1, trip 1-39	Fault bus 39, trip 1-39	Fault bus 3, trip 3-4	Fault bus 4, trip 3-4	Fault bus 14, trip 14-15	Fault bus 15, trip 14-15	Fault bus 15, trip 15-16	Fault bus 16, trip 15-16	Fault bus 16, trip 16-17	Fault bus 17, trip 16-17
No. of secure instances	3257	3075	3417	3326	3419	3462	3394	3437	3320	3282
No. of insecure instances	1786	1968	1626	1717	1624	1581	1649	1606	1723	1761

TABLE IV
GAN PERFORMANCE

PMU Options	Average Computation Efficiency	Filling Missing Data Performance (MAPE Under Different Missing PMU Numbers)											
		1	2	3	4	5	6	7	8	9	10	11	12
I	1.97ms	1.29%	1.55%	1.71%	1.98%	2.11%	2.27%	2.58%					
II	2.24ms	1.33%	1.42%	1.56%	1.73%	1.79%	1.84%	2.01%	2.06%	2.25%	2.28%	2.55%	2.86%

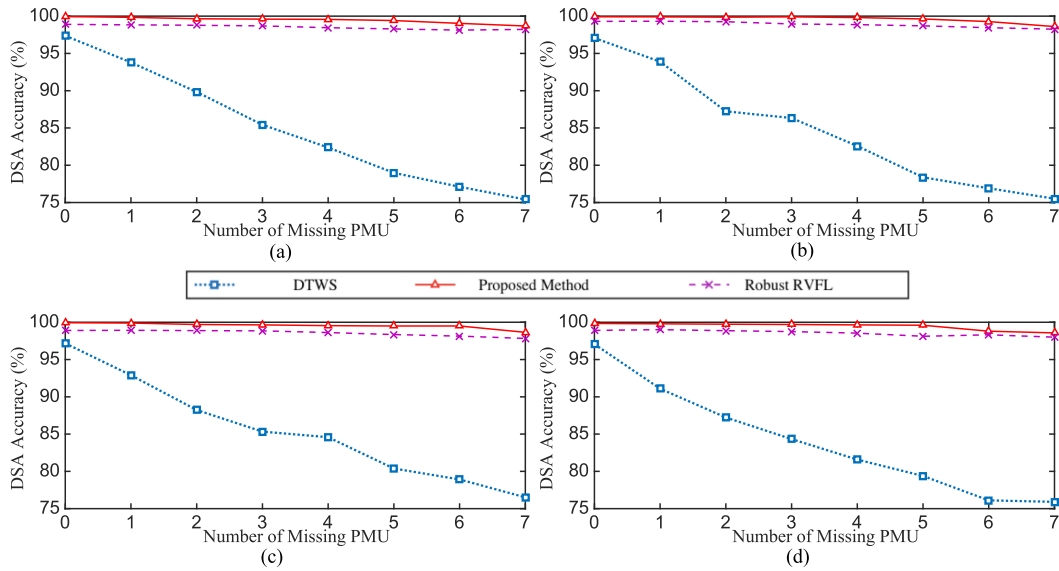


Fig. 6. DSA testing results of proposed method, robust RVFL ensemble and DTWS under the PMU placement option I. (a) Average 10 faults. (b) Fault 1. (c) Fault 5. (d) Fault 10.

and feature estimation [7]. Note that all the results are the average value of the 10 test runs for the ten contingencies.

1) *Performance on Filling up Missing Data*: Firstly, the GAN's performance on filling up the missing data is tested. The error is evaluated by mean average percentage error (MAPE) [24]:

$$\text{MAPE} = \frac{1}{N_t} \cdot \sum_{i=1}^{N_t} \left| \frac{y_i^{\text{real}} - y_i^{\text{predicted}}}{y_i^{\text{real}}} \right| \quad (10)$$

where N_t is the total number of the testing instances, y_i^{real} and $y_i^{\text{predicted}}$ are the real data value and predicted missing data value, respectively.

The results in Table IV are the MAPE for two PMU placement options under all missing PMU scenarios. It can be seen that the prediction of missing data is very close to the real value, which demonstrates that the generated data by the GAN model can accurately fill up the missing data. Moreover, the average computation time are 1.97 ms and 2.24 ms of one instance under two different PMU placement locations respectively, which are negligible for online DSA application.

2) *Classification Accuracy Comparison*: The proposed method is compared with two existing methods, robust RVFL ensemble learning [6] and DTWS [4]. The testing results corresponding to PMU placement options I–II are demonstrated in Fig. 6(a–d) and Fig. 7(a–d), where DSA accuracy is the percentage of the testing samples that are correctly classified, and each

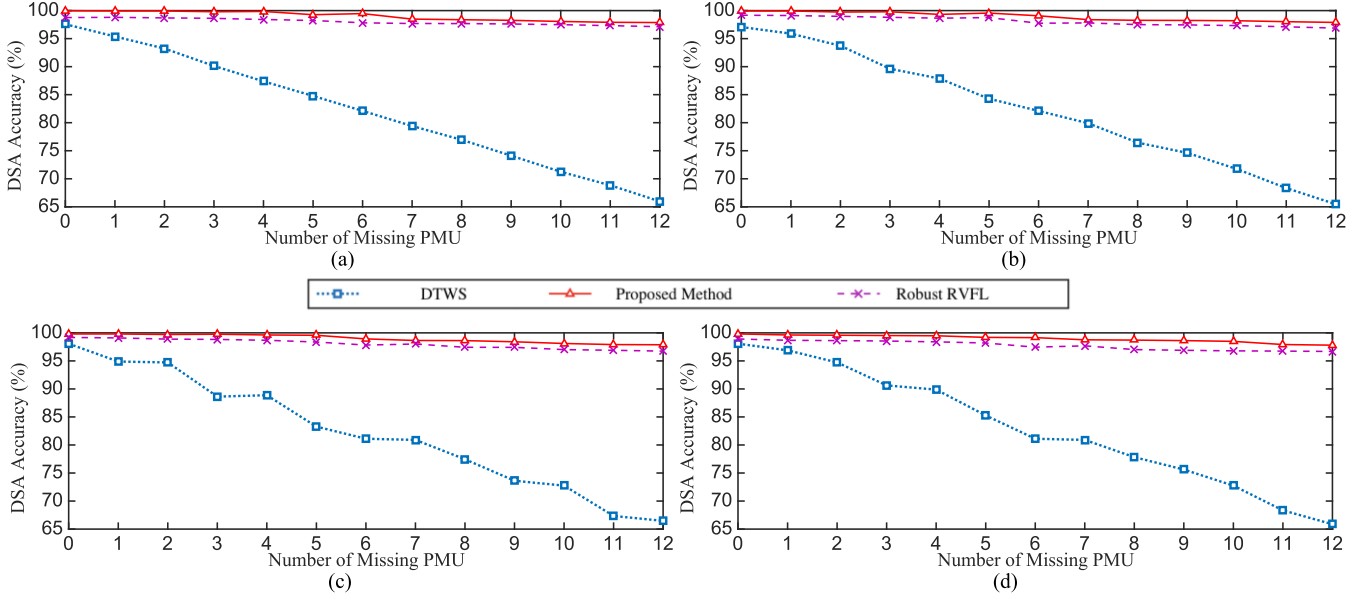


Fig. 7. DSA testing results of proposed method, robust RVFL ensemble and DTWS under the PMU placement option II. (a) Average 10 faults. (b) Fault 1. (c) Fault 5. (d) Fault 10.

TABLE V
ADAI RESULTS AND COMPUTATIONAL EFFICIENCY OF DIFFERENT METHODS

Method	Computational Efficiency (No. of Classifiers)		ADAI	
	PMU Option I	PMU Option II	PMU Option I	PMU Option II
Proposed Method	1	1	99.40%	99.04%
Robust Ensemble Learning [6]	19	36	98.48%	97.96%
DTWS Method [4]	1	1	83.28%	80.81%
Feature Estimation [7]	255	8191	96.99%	96.12%

point represents the average DSA accuracy under each identical number of missing random PMUs data. Fig. 6(a) and Fig. 7(a) show the average DSA accuracy under 10 typical three-phase faults, Fig. 6(b-d) and Fig. 7(b-d) are randomly selected to show the DSA accuracy of the fault 1, 5, 10 under the PMU placement options I–II, respectively. According to Fig. 3 and Fig. 4, with the increase of the number of missing data, robust ensemble learning and DTWS show the different degree of decline in accuracy. By contrast, the proposed model is not significantly influenced, and still can maintain a relatively high DSA accuracy.

3) *Average Accuracy Against PMU Missing Comparison:* In this paper, average DSA accuracy index (ADAI) is utilized to quantify the DSA accuracy and robustness for incomplete PMU measurement:

$$ADAI = \frac{1}{M} \frac{1}{F_c} \sum_{f=1}^{F_c} \sum_{m=1}^M A_m \times 100\% \quad (11)$$

where F_c represents the number of different contingencies. A_m is the average DSA accuracy with m PMU missing, ADAI is the average DSA accuracy under all PMU missing conditions.

A larger ADAI value implies higher DSA accuracy and robustness against incomplete PMU measurements. Two

state-of-the-art methods are also compared in Table V. It can be seen that the proposed method always has the highest accuracy and thus robustness against missing data under the PMU placement options I–II among three methods.

4) *Computational Efficiency Comparison:* Another significant advantage is the computational efficiency. The proposed method can fill up the incomplete PMU data under any PMU missing scenario by training only one classifier. But the existing methods need to train more classifiers to be practical, for N PMUs, the total number of classifiers could be $2^N - 1$ at most. The number of classifiers needed to perform the DSA is also compared in Table V, where it can be seen that the proposed method is much computational efficient than the existing methods which requires many more classifiers to be trained.

V. CONCLUSION

In this paper, a fully data-driven method based on DSA model and GAN model is developed which aims to maintain the DSA accuracy and the system stability under any PMU missing conditions without depending on PMU observability. GAN model can complete the missing data maintaining the original feature characters, and the DSA model can make the decision with the

complete data. As mathematical formula, it can ensure that GAN reach the global optimization when the discriminator cannot justify the generated data and historical data. The proposed IS is tested on the New England 10-machine 39-bus system.

Being fully data-driven, the proposed method does not depend on the PMU observability and network topologies, during online application, system operators only need to input the practical PMU measurements without considering the detail missing measurements information to select the model or generated data. Therefore, the proposed method offers the system operators more flexibility in manipulating the assessment performance, which makes the proposed IS more practical in real-world assessment against the PMU missing conditions. By marked contrast, the state-of-the-art methods have the much higher computational complexity, the proposed method can only utilize one classifier to maintain a high DSA accuracy.

REFERENCES

- [1] Z. Y. Dong, Y. Xu, P. Zhang, and K. P. Wong, "Using IS to assess an electric power system's real-time stability," *IEEE Intell. Syst.*, vol. 28, no. 4, pp. 60–66, Jul./Aug. 2013.
- [2] Y. Xu, Z. Y. Dong, J. H. Zhao, P. Zhang, and K. P. Wong, "A reliable intelligent system for real-time dynamic security assessment of power systems," *IEEE Trans. Power Syst.*, vol. 27, no. 3, pp. 1253–1263, Aug. 2012.
- [3] F. Luo *et al.*, "Advanced pattern discovery-based fuzzy classification method for power system dynamic security assessment," *IEEE Trans. Ind. Inform.*, vol. 11, no. 2, pp. 416–426, Apr. 2015.
- [4] T. Guo and J. V. Milanovic, "The effect of quality and availability of measurement signals on accuracy of on-line prediction of transient stability using decision tree method," in *Proc. 4th Innovative Smart Grid Technol. Eur.*, 2013, pp. 1–5.
- [5] M. He, V. Vittal, and J. Zhang, "Online dynamic security assessment with missing PMU measurements: A data mining approach," *IEEE Trans. Power Syst.*, vol. 28, no. 2, pp. 1969–1977, May 2013.
- [6] Y. Zhang, Y. Xu, and Z. Y. Dong, "Robust ensemble data analytics for incomplete PMU measurements-based power system stability assessment," *IEEE Trans. Power Syst.*, vol. 33, no. 1, pp. 1124–1126, Jan. 2018.
- [7] Q. Li, Y. Xu, C. Ren, and J. Zhao, "A hybrid data-driven method for online power system dynamic security assessment with incomplete PMU measurements," in *Proc. IEEE Power Energy Soc. Gen. Meeting, USA*, Aug. 2019.
- [8] Y. Zhang, Y. Xu, and Z. Y. Dong, "Robust classification model for PMU-based on-line power system DSA with missing data," *IET Gen. Transmiss. Distrib.*, vol. 11, pp. 4484–4491, Dec. 2017.
- [9] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
- [10] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 2536–2544.
- [11] Y. Chen, Y. Wang, D. Kirschen, and B. Zhang, "Model-free renewable scenario generation using generative adversarial networks," *IEEE Trans. Power Syst.*, vol. 33, no. 3, pp. 3265–3275, May 2018.
- [12] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–41.
- [13] W. F. Schmidt M. A. Kraaijveld, and R. P. W. Duin, "Feedforward neural networks with random weights," in *Proc. 11th Int. Conf. Pattern Recognit. Vol. II. Conf. B, Pattern Recognit. Methodology Syst.*, 1992, pp. 1–4.
- [14] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: A new learning scheme of feedforward neural networks," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2004, vol. 2, pp. 985–990.
- [15] Y. H. Pao and S. M. Phillips, "The functional link net and learning optimal control," *Neurocomputing*, vol. 9, no. 2, pp. 149–164, 1995.
- [16] G.-B. Huang *et al.*, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006.
- [17] Y. Ren, L. Zhang, and P. N. Suganthan, "Ensemble classification and regression—recent developments, applications, and future directions," *IEEE Comput. Intell. Mag.*, vol. 11, no. 1, pp. 41–53, Feb. 2016.
- [18] Z.-H. Zhou *et al.*, "Ensembling neural networks: Many could be better than all," *Artif. Intell.*, vol. 137, no. 1/2, pp. 239–263, 2002.
- [19] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 10, pp. 993–1001, Oct. 1990.
- [20] C. Ren, Y. Xu, Y. Zhang, and C. Hu, "A multiple randomized learning based ensemble model for power system dynamic security assessment," in *Proc. IEEE Power Energy Soc. Gen. Meeting, Portland, OR, USA*, Aug. 2018, pp. 1–5.
- [21] K. Meng, Z. Dong, K. Wong, Y. Xu, and F. Luo, "Speed-up the computing efficiency of power system simulator for engineering-based power system transient stability simulations," *IET Gen. Transmiss. Distrib.*, vol. 4, no. 5, pp. 652–661, 2010.
- [22] S. Chakrabarti and E. Kyriakides, "Optimal placement of phasor measurement units for power system observability," *IEEE Trans. Power Syst.*, vol. 23, no. 3, pp. 1433–1440, Aug. 2008.
- [23] Y. Xu, Z. Y. Dong, K. Meng, R. Zhang and K. P. Wong, "Real-time transient stability assessment model using extreme learning machine," *IET Gen. Transmiss. Distrib.*, vol. 5, no. 3, pp. 314–322, Mar. 2011.
- [24] J. S. Armstrong and F. Collopy, "Error measures for generalizing about forecasting methods: Empirical comparisons," *Int. J. Forecasting*, vol. 8, no. 1, pp. 69–80, 1992.
- [25] R. Zhang, Y. Xu, Z. Y. Dong, and K. P. Wong, "Post-disturbance transient stability assessment of power systems by a self-adaptive intelligent system," *IET Gen. Transmiss. Distrib.*, vol. 9, no. 3, pp. 296–305, Feb. 2015.



Chao Ren (S'18) received the B.E. degree from the School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2017. He is currently working toward the Ph.D. degree with the Interdisciplinary Graduate School, Nanyang Technological University, Singapore. His research interests include machine learning, data analytics, optimization, computational intelligence, and its applications to power engineering.



Yan Xu (S'10–M'13–SM'19) received the B.E. and M.E. degrees from the South China University of Technology, Guangzhou, China, in 2008 and 2011, respectively, and the Ph.D. degree from The University of Newcastle, Callaghan, NSW, Australia, in 2013. He is currently the Nanyang Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. Previously, he held The University of Sydney Post-doctoral Fellowship in Australia. His research interests include power system stability and control, microgrid, and data-analytics for smart grid applications. He is an Editor for the IEEE TRANSACTIONS ON SMART GRID and CSEE Journal of Power and Energy Systems, and an Associate Editor for IET Generation, Transmission, and Distribution.