# A Deep Learning-Based Feature Extraction Framework for System Security Assessment

Mingyang Sun, *Member, IEEE,* Ioannis Konstantelos, *Member, IEEE,* Goran Strbac, *Member, IEEE*

***Abstract*—The ongoing decarbonisation of modern electricity systems has led to a substantial increase of operational uncertainty, particularly due to the large-scale integration of renewable energy generation. However, the expanding space of possible operating points renders necessary the development of novel security assessment approaches. In this paper we focus on the use of security rules, where classifiers are trained offline to characterize previously unseen points as safe or unsafe. This paper proposes a novel deep learning-based feature extraction framework for building security rules. We show how deep autoencoders can be used to transform the space of conventional state variables (e.g. power flows) to a small number of dimensions where we can optimally distinguish between safe and unsafe operation. The proposed framework is data-driven and can be useful in multiple applications within the context of security assessment. To achieve high accuracy, a novel objective-based loss function is proposed to address the issue of imbalanced safe/unsafe classes that characterizes electricity system operation. Furthermore, an R-vine copula-based model is proposed to sample historical data and generate large populations of anticipated system states for training. The superior performance of the proposed framework is demonstrated through a series of case studies and comparisons using the load and wind generation data from the French transmission system, which have been mapped to the IEEE 118-bus system.**

***Index Terms*—Deep learning, feature extraction, Monte Carlo simulation, R-vine copulas, security assessment.**

## I. Introduction

In modern electricity systems, the large-scale integration of intermittent energy sources and the potential electrification of the transport and heat sectors [1] are significantly expanding the operating state-space of electricity systems [2]. At the same time, electricity market liberalization has largely unbundled the distribution and supply services in many jurisdictions, reducing the system controllability for system operators. Consequently, the afore-mentioned aspects bring about unprecedented challenges in enabling the stable and reliable operation of electricity systems. For Transmission System Operators (TSOs), the power system security assessment is of paramount importance for determining safe system operating boundaries [3]. There are many different criteria to define power grid security; static insecurity could refer to line overloading, voltage exceeding limits whereas dynamic insecurity could be generator rotor angle instability or voltage instability. In this paper, although we focus on the criterion of load curtailment, the proposed deep learning-based framework could be readily extended to dynamic security assessment of power systems.

M. Sun, I.Konstantelos, and G. Strbac are with the Department of Electrical & Electronic Engineering, Imperial College London, London, SW7 2AZ, UK.

In the literature, system security assessment methods can be categorized into analytical physics approaches and data-driven approaches. Specifically, the time-domain simulation of the nonlinear differential equations that model the power system is the most straightforward analytical approach [4]. However, detailed information on network configuration during and after a fault are required, which entails a massive computational load [5]. Moreover, an alternative analytical physics approach, transient-energy-function method, is conducted based on Lyapunov stability or Energy Function principle [6] with the difficulties in determining the levels of kinetic and potential energy under a given set of faults in the context of practical power systems. According to these aforementioned limitations, dynamic stability assessment across a host of operating points and against a large set of potential contingencies is intractable for a large-scale power system. To this end, data-driven approaches including curve fitting techniques (e.g.,[7]) were proposed so that the network configuration information and power system parameters are not required to predict transient instabilities. Nevertheless, the curve-fitting methods exhibit a low prediction accuracy caused by their characteristics of being sensitive to the start-up time of prediction and the sampling period [5].

In recent years, with the widespread adoption of Phasor Measurement Units (PMUs), an influx of real data from system's past history provides valuable opportunities to construct more reliable system security rules via artificial intelligence methods instead of using conventional heuristic rules. In the literature, machine learning techniques such as Support Vector Machines (SVMs) [8], artificial neural networks (ANN)[9], decision trees (DTs) [10], and long short-term memory (LSTM) networks [11], have been widely employed to build security rules for transient stability. Although different varieties of advanced classifiers have been proposed to improve prediction accuracy, the derived security rules have sometimes been counter-intuitive, based on classifying features that appear to be largely unrelated to the fault analyze. These variables may be chosen due to spurious correlations observed in the data. In other cases, rules may correctly relate the natural relations between variables, but produce incorrect results when used to design preventive or corrective control actions. To this end, feature selection can be conducted to prevent spurious correlations and identify the key discriminating variables.

Feature selection is defined as the process of identifying the features that contribute most to the discrimination ability of a classifier [12]. In general, two approaches can be considered for improving feature selection. The first approach is data-driven: the application of sparse feature selection methods

such as random feature selection random forests, LASSO, etc. as part of the machine learning training phase. The second approach takes into account the underlying physics of the problem, for instance by considering only classifying variables that pertain to physical assets in the geographic area of the contingency. Hybrid approaches that combine data-driven and heuristic knowledge of the system can also be developed. Beyond the feature selection approaches, which need extensive domain expertise and careful engineering, feature extraction can be considered as an alternative way to extract representative features from the input data in an automatic or nearly automatic fashion. Unlike feature selection, feature extraction aims to map raw input features into a lower-dimensional space that is more discriminative for classifiers. Conventional linear feature extractors (e.g., principal component analysis (PCA), linear discriminant analysis (LDA) have been demonstrated as useful methods but with limited performance due to the linear transformation procedure. To this end, a series of nonlinear feature extraction methods have been proposed in the literature such as kernel PCA [13] and GPLVM [14].

As one of the cutting-edge approaches, deep learning aims to learn representations of data with multiple levels of abstractions by using multiple layers of computational models [15]. Although the concept of deep learning has been proposed for decades, the performance of deep learning was limited by the training approach, the insufficient data, and the computational power. Nowadays, the aforementioned challenges are being gradually addressed with the introduction of new techniques, the influx of high-resolution and high-quality data, and the development of high-performance computing [16]. Although the superior performance of deep learning approaches has been demonstrated in a wide range of areas (e.g., speech recognition, objective detection, pattern recognition, image processing, etc.), very limited work have been done to solve power system problems, especially for extracting features for system security assessment. Most of the the state-of-the-art work focus on exploiting deep learning techniques to improve the performance of short-term household load forecasting[16], RES output forecasting [17], socio-demographic information identification[18] and real-time detection of false data injection attacks in smart grid [19]. The success of the afore-mentioned applications can be attributed to its powerful capability of learning high-level representations of raw input data. To this end, it is important and imperative to exploit such power technique to address the challenges of system security assessment from the aspect of extracting effective features.

After determining a powerful feature extractor, another crucial topic is: *How to provide sufficient and effective training data for deep learning to enable a accurate and reliable result?* This question is raised due to the fact that the performance of deep learning is highly dependent on the quantity and quality of training data. In other words, deep Learning algorithms are quite beneficial when dealing with learning from large amounts of data [20], [21]. As illustrated in [22], [23], a parametric model is capable of generating training databases of arbitrarily large size that are similar but not identical to what has already been encountered, effectively interpolating and extrapolating the historical datasets. In general, modeling and sampling high-dimensional stochastic variables suffer from four key challenges: 1) high-dimensionality; 2) the large number of observations; 3) non-Gaussian marginal probability distribution of each variable; and 4) the non-linear complex dependency structure. To deal with these issues, a novel composite approach is proposed in [23] to model and sample high-dimensional stochastic variables in power systems based on the stages of clustering, dimensionality reduction, and vine-copulas modeling. In particular, their considered C-vine and D-vine copulas exhibited superior performance in capturing sophisticated dependency structure among the vast number of stochastic variables.

When training a deep learning-based feature extractor, supervised fine-tunning with backpropagation is performed to maximize the separability of the extracted features while minimizing the classification error. It is imperative to note that the security classification problem is actually an imbalanced data classification problem because unsafe states only account for a very small proportion of the whole training dataset. Without explicitly considering this fact, the illusion of high accuracy may neglect rare but severe cases and misleads the feature extraction with a counterproductive loss function in the fine-tuning procedure. Many conventional methods have been proposed to address this issue, which can be typically categorized into sampling methods, cost-sensitive methods, and an ensemble of classifiers [24].

In this paper, we first propose a novel deep learning-based feature extraction framework for system security assessment. R-vine copula-based sampling strategy is employed to enrich the training database for deep learning while capturing the complex non-linear dependency structure among high-dimensional stochastic variables of loads and wind power injections. An objective-based loss function is proposed to deal with the issue of imbalanced data. A comprehensive comparison is conducted between different methods to illustrate the benefits of the proposed approach. It is important to highlight that this work focuses on proposing a novel feature extractor, rather than a new classifier, which can construct effective features for more potential applications such as stochastic variables sampling, scenario reduction, key features detection, data compression, etc. The key contributions of this paper can be summarized as follows:

1) A novel deep learning-based feature extraction framework for system security assessment is proposed, for the first time, to automatically extract effective training features that can be readily categorized by classifiers. Additionally, the proposed model is further developed to predict the security status for multiple contingencies;

2) An objective-based loss function is presented, specifically aimed at addressing the issue of imbalanced classes that arises in the context of security assessment.

3) An R-vine copula-based modeling and sampling framework is implemented to enrich the population of anticipated system states and improve the effectiveness of the proposed deep neural network.

The remainder of this paper is organized as follows. Section II introduces the framework and provides the technical details. Section III illustrates the sampling method based on R-vine

copulas. Section IV introduces the proposed deep autoencoder networks. Section V illustrates the considered evaluation metrics. Section VI conducts numerical experiments on the IEEE 118-bus system. Finally, the conclusion is presented in Section VII.

## II. Proposed Deep Learning-Based Feature Extraction Framework

The deep learning-based feature extraction framework proposed in this paper is illustrated in Fig. 1. First, the composite modeling approach proposed in [23] is further developed by using a more flexible graphical model, regular vine (R-vine), which can capture more complex dependence structure among stochastic variables. After running the pre-fault and post-fault simulations given a set of contingencies, deep autoencoder network with the proposed objective-based loss function is employed to extract important features that can effectively reduce the difficulty in classifying the stable and unstable operating points. Note that in this case "simulation" refers to solving an OPF problem to achieve the indicators for supervised learning. Specifically, the framework includes the following four main stages:

1) ***Sampling Operating Points***: This stage aims to enrich the training database for the deep learning-based feature extractor via constructing a R-vine copula model and then generating a sufficient number of samples. Given the historical stochastic variables $X \in \mathbb{R}^{T\times M}$, where $T$ and $M$ denote the numbers of measurements and variables, respectively, the first step is to partition the observations into training set $X_{train} \in \mathbb{R}^{T_{train}\times M}$ and test set $X_{test} \in \mathbb{R}^{T_{test}\times M}$, where

$$T_{train} \approx 80\% \times T, \tag{1}$$

$$T_{test} = T - T_{train}. \tag{2}$$

Then clustering is first performed to partition the observations of $X_{train}$ into K clusters where $X_k \subset X_{train}$ for $k = 1, ..., K$. For each cluster, empirical cumulative distribution functions (ECDFs) are used to transform the data from original domain to the rank-uniform domain $U_k$. Then dimensionality reduction is performed on each $U_k \in \mathbb{R}^{T_k\times M}$ to get the low-dimensional data $L_k \in \mathbb{R}^{T_k\times q_k}$ where $q_k < M$.

In order to build the R-vine copulas model, the uniform transformation is performed again for each $L_k$ to obtain $U_k^L$ in the $[0,1]^{q_k}$ domain. Based on $U_k^L$, the sequential method is performed to determine the optimal R-vine specification matrix $S_k \in \mathbb{R}^{q_k\times q_k}$, the best-fit bivariate copula families $B_k \in \mathbb{R}^{(q_k-1)\times(q_k-1)}$, and the estimated parameters $\Theta_k \in \mathbb{R}^{(q_k-1)\times(q_k-1)}$. Given the total number of samples $T^s$, the number of samples $T_k^s$ for each cluster can be calculated by

$$T_k^s = T^s \times W_k, \tag{3}$$

where $W_k = |X_k| / |X_{train}|$. Simulating the $K$ constructed R-vine models individually and generate the samples $\hat{X}_K \in \mathbb{R}^{T_k^s\times q_k}$. Finally, after a series of back-projection procedures as described in [23], the output of this stage is

$$\hat{X} = [\hat{X}_1, ..., \hat{X}_K] = \{\vec{\hat{x}}_t, t = 1, ..., T_s\} \in \mathbb{R}^{T_s\times M}. \tag{4}$$

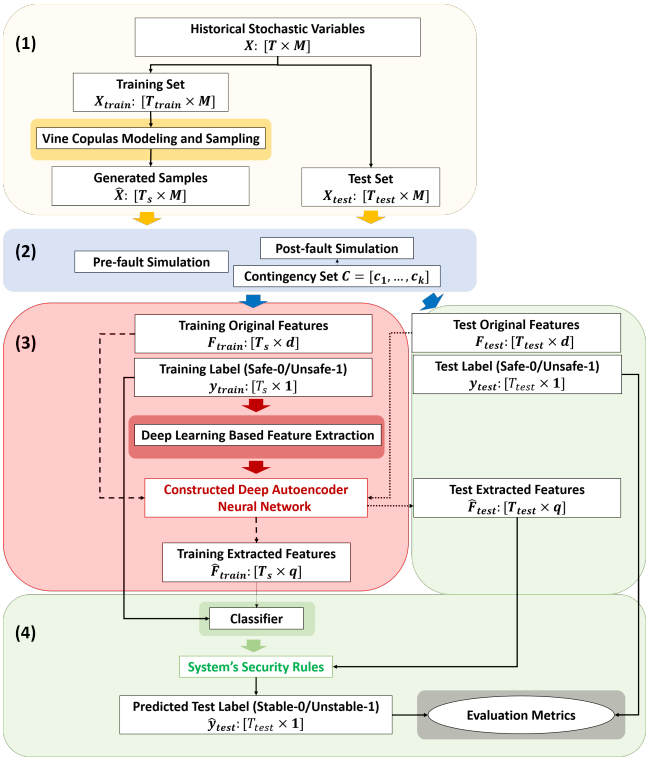



Fig. 1. The proposed deep learning based feature extraction and system security assessment framework.

2) ***Pre-fault and Post-fault Simulations***: In this stage, pre-fault and post-fault simulations are performed based on the generated samples to construct the features and labels of the training and test datasets, respectively. Let $N$, $L$, $G$ denote the sets of network buses, transmission lines, and generators, receptively. For each sampled operating point $\vec{\hat{x}}_t$ in $\hat{X}$ the DC Optimal Power Flow problem is solved for the optimal dispatch schedule $ods_t$. Based on the sampled training set $\hat{X}$, the output of the pre-fault simulation is a set of original training features $F_{train} \in \mathbb{R}^{T_s\times d}$, where $d$ is the total number of original training features, consisting of loads $X^D \in \mathbb{R}^{T_s\times|N|}$, power injections $I^P \in \mathbb{R}^{T_s\times|N|}$, power flows $F^L \in \mathbb{R}^{T_s\times|L|}$, phase angles $\theta \in \mathbb{R}^{T_s\times|N|}$, and generation outputs $P \in \mathbb{R}^{T_s\times|G|}$, expressed as follows:

$$F_{train} = [X^D, I^P, F^L, \theta, P]. \tag{5}$$

Subsequently, post-fault simulation is carried out for each schedule under each contingency $c_i$ in $C = [c_1, ..., c_l]$. As such, the *safe/unsafe* labels

$$y_{train} = [y_{train}^{c_1}, ..., y_{train}^{c_l}] \in \mathbb{R}^{(l\times T_s)\times 1} \tag{6}$$

are assigned to the system post-fault states. Note that *safe* represents that post-fault load does not have curtailment, whereas *unsafe* indicates the existence of load curtailment under contingency $c$. In addition, one-hot encoding is used to construct the labels of contingencies

$$L^C \in \mathbb{R}^{(l\times T_s)\times l} \tag{7}$$

which are considered as additional training features. Therefore, the final original training features $F_{train}$ =

$[X^D, I^P, F^L, \theta, P, L^C] \in \mathbb{R}^{(l\times T_s)\times(d+l)}$. Following the above procedure, we can also obtain the original test features $F_{test} \in \mathbb{R}^{(l\times T_{test})\times(d+l)}$ and their corresponding labels $y_{test} \in \mathbb{R}^{(l\times T_{test})\times 1}$.

3) ***Deep Autoencoder Based Feature Extraction***: As the key stage of the proposed framework, deep autoencoder network is constructed by performing pretraining, unrolling, and supervised fine-tuning. Detailed information are provided in Section III. Let $q$ denote the number of neurons of the last hidden layer, the output of this stage is the constructed deep autoencoder neural network. Given the original training and test features, the extracted training features $\hat{F}_{train} \in \mathbb{R}^{(l\times T_s)\times q}$ and the extracted test features $\hat{F}_{test} \in \mathbb{R}^{(l\times T_{test})\times q}$ can be obtained through the constructed network. Note that the hyperparameters such as learning rate, number of layers, size of each layer (i.e., include the dimension $q$ of the output extracted features), class weights, are determined via grid search and cross validation [18].

4) ***Security Rules Construction***: Finally, security rules are constructed in this stage. Based on the extracted features $\hat{F}_{train}$, a ten-fold validation scheme is employed to construct the system's security rules by using a tested classifier. Afterwards, the performance of the built rules can be assessed by calculating the evaluation metrics on the basis of the predicted security status $\hat{y}_{test}$ of $\hat{F}_{test}$ and the actual security status $y_{test}$.

To summarize, the proposed deep learning based feature extraction framework is outlined in Algorithm 1.

**Algorithm 1** Deep Learning Based Feature Extraction and Security Assessment Framework

**Input:** Historical load and wind generation data: $X$, Number of clusters: $K$, Information retainment threshold: IR, Number of samples: $T_s$, Contingency set: $C$, Total number of layers: $L$, Size of each layer: $N_L$ .

**Output:** Sampled load and wind data: $\hat{X}$, Deep Autoencoder Network: DAN$_f$, Security Assessment Model: DT$_s$, Evaluation Metrics: $E$

***Step 1****: Partition the observations of X into training set $X_{train}$ and test set $X_{test}$. Construct the proposed R-vine copulas model based on $X_{train}$ and then generate $T_s$ samples $\hat{X}$.*

1: $\hat{X} = RVine(X_{train}, K, IR, T_s)$.

***Step 2****: Run the pre-fault simulation and post-fault simulation with contingency set C to obtain the original features and the corresponding labels.*

2: $[F_{train}, F_{test}]$ = $Pre\text{-}fault(\hat{X}, X_{test})$.

3: $[y_{train}, y_{test}]$ = $Post\text{-}fault(\hat{X}, X_{test}, C)$.

***Step 3****: Given the number of layers L and sizes of layers $N_L$, construct the deep learning based feature extractor model DAN$_f$ based on $F_{train}$ and $y_{train}$.* In addition, the extracted features $\hat{F}_{train}$ can be obtained in this step.

4: [$\hat{F}_{train}$, DAN$_f$] = $DAN(F_{train}, y_{train}, L, N_L)$.

***Step 4****: Train the classier DT based on $\hat{F}_{train}$ and $y_{train}$ and then output the constructed security assessment model. Obtain the extracted test features $\hat{F}_{test}$ and then predict the corresponding label $\hat{y}_{test}$. Finally, assess the performance of the proposed framework by calculating the evaluation metrics E.*

5: DT$_s$=$DecisionTree(\hat{F}_{train}, y_{train})$.

6: $\hat{y}_{test}$=DT$_s(\hat{F}_{test})$.

7: $E$ = ConfusionMatrix$(\hat{y}_{test}, y_{test})$.

## III. Sampling Using R-vine Copulas

In this paper, the probability density function and the cumulative distributions function as well as their copula versions are defined as $f$, $F$, $c$ and $C$, respectively. Given $M$ random variables $X = [X_1, ..., X_M] \in \mathbb{R}^{T\times M}$, based on Sklar's theorem [25], the joint probability density function $f(X_1, ..., X_M)$ can be expressed as $(\prod_{i=1}^{M} f_i(X_i)) \times c_{1...M}(F_1(X_1), ...F_M(X_M))$, where the copula density function $c_{1...M} : [0,1]^M$ describes the dependence structure among uniform random variables $\{U_1, ..., U_M\} = \{F_1(X_1), ..., F_M(X_M)\}$. Note that we use the empirical cumulative distribution function (ECDF) and its inverse version ECDF$^{-1}$ to model the non-Gaussian marginal distributions of $X$. In general, there are various types of copula families that can be employed to model different complex dependence structures. However, most of them are limited to a bivariate version and it becomes more effective to decompose a multivariate copula into the product of a cascade of bivariate copulas, denoted the pair-copula construction (PCC) [26]. As such, PCC can introduce flexibility in capturing more complex dependence structure among loads and wind generation outputs.

As one of the most flexible graphical vine copula models, regular vine (R-vine), it is composed of a nested set of $M-1$ trees $\Upsilon = (T_1, ..., T_{M-1})$ such that the edges $E_j$ of tree $T_j$ become the nodes $N_{j+1}$ of tree $T_{j+1}$, for $j = 1, ..., M-1$. As defined in [27], $\Upsilon$ of an R-vine on $M$ random variables is required to satisfy the following conditions: i) the first tree $T_1$ contains a set of edges $E_1$ and nodes $N_1 = 1, ..., M$; ii) for $i = 2, ..., M$, $T_i$ consists of edges $E_i$ and nodes $N_i = E_{i-1}$; and iii) for $i = 2, ..., M-1$, $\{j, k\} \in E_i$ must hold that $\#(j \cap k) = 1$, where $j = \{j_1, j_2\}$ and $k = \{k_1, k_2\}$.

For a regular vine $\Upsilon$, let $S_e = \{\nu \in N_1 | \exists e_i \in E_i, i = 1, ..., m_g - 1$,with $\nu \in e_1 \in ... \in e_{m_g-1} \in e\}$ denoting the complete union of an edge $e = \{j, k\} \in E_q$ in tree $T_q$. The conditioning set and the conditioned sets associated with edge $e = \{j, k\}$ are defined as $\Psi_e := S_j \cap S_k$ and $\{\Omega_{e,j} = S_j \backslash \Psi_e$, $\Omega_{e,k} = S_k \backslash \Psi_e\}$, respectively, where $(-)\backslash(*) := (-) \cap (*)^C$ and $(*)^C$ is the complement of $(*)$. Following the above-mentioned definitions, the density function $f(X_1, ..., X_M)$ can be decomposed as follows:

$$\prod_{i=1}^{M} f_i(X_i) \times \prod_{i=1}^{M-1} \prod_{e\in E_i} c_{\Omega_{e,j},\Omega_{e,k}|\Psi_e}(F_{\Omega_{e,j}|\Psi_e}(.), F_{\Omega_{e,k}|\Psi_e}(.)) \tag{8}$$

where $e = \{j, k\}$ and $F_{\Omega_{e,j}|\Psi_e}(.) = F_{\Omega_{e,j}|\Psi_e}(X_{\Omega_{e,j}}|X_{\Psi_e})$ which is denoted as an $h$-function. The detailed formulation of $h$-function for R-vine is presented in [27]. In addition, to select the most appropriate nodes-edges-trees combination for R-vine, the sequential method was proposed in [28] as an automated strategy to select the R-vine tree that maximizes the sum of absolute empirical Kendall's $\tau$. Regarding the computational complexity, the number of possible R-vines

rapidly increases with $M$, thus leading to an impractical issue for very high-dimensional cases. To this end, as proposed in [23], hierarchical clustering with average linkage and locality preserving projections (LPP) can be employed to accelerate the modeling procedure via a series of domain transformations and reconstructions. Note that in this paper the candidate pair copula families include Gumbel, Frank, Clayton, Gaussian, Student-t, BB1, BB6, BB7, and BB8 as well as their 90°, 180°, and 270° rotated versions. The dependence structure of historical load and wind generation data can be well-captured by employing such varieties of copula families.

Regarding the number of samples, in general, higher-resolution data (e.g., 5 minutes time interval) with a larger number of samples may help to train a more effective classifier for system security assessment. However, the massive amount of data may lead to computational load issues. To this end, a trade-off between prediction accuracy and computing time need to be made when determining the number of generated samples. Moreover, the appropriate size and type of data need to be determined according to the complexity of the system studied and the number of stochastic variables that need to be modeled. For high-dimensional stochastic variables, more data need to be sampled from the constructed statistical model to accurately represent the complex dependence structure and include more unseen but possible system states for the classifier to train. It is imperative to note that classifier training is actually an offline process and therefore, training an effective classifier with high accuracy is the primary task. On the other hand, the computational issue could be alleviated by using high performance cloud computing, which has been increasingly employed in industry to handle the big data problem.

## IV. The Proposed Deep Autoencoder Networks

### A. Autoencoder

Traditional autoencoder is an unsupervised artificial neural network including a visible layer, a hidden layer, and a reconstruction layer that is trained to learn a representation for the input data set while minimizing the difference between the input original and output reconstructed data sets [29]. Let

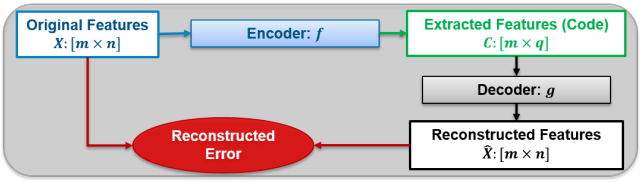



Fig. 2. Traditional unsupervised autoencoder.

$X = [\vec{x}_1, ..., \vec{x}_m]^T \in \mathbb{R}^{m \times n}$ denote the input data where $X$ refers to $F_{train}$, $m = l \times T_s$ and $n = d+l$, as defined in Section II, an autoenocder network consists of an encoder function $C = f(X)$ and a decoder that produces a reconstruction $\tilde{X} = g(C)$, where $C = [\vec{c}_1, ..., \vec{c}_m]^T \in \mathbb{R}^{m \times q}$ and $\tilde{X} = [\tilde{x}_1, ..., \tilde{x}_m]^T \in \mathbb{R}^{m \times n}$. Then the objective of conventional autoencoder can be described as extracting effective features representations $C$ of $X$ and minimizing the reconstructed error between $X$ and $\tilde{X}$, as illustrated in Fig. 2. In particular, for each input vector $\vec{x}_i \in \mathbb{R}^n$, a hidden representation $\vec{c}_i \in \mathbb{R}^q$ can be obtained through its hidden layer activation function:

$$\vec{c}_i = s(W\vec{x}_i + b) \tag{9}$$

where $s$ is the nonlinear sigmoid activation function $s(z) = \frac{1}{1+\exp(-z)}$, $b \in \mathbb{R}^n$ is a bias vector for visible layer, and $W \in \mathbb{R}^{q \times n}$ is a weight matrix that associates the visible layer and hidden layer. Then the reconstruction $\tilde{x}_i$ can be calculated as

$$\tilde{x}_i = s(W^T\vec{c}_i + b^T) \tag{10}$$

where $W^T$ is the reconstruction weight matrix and $b^T$ is the reconstruction bias vector. The reconstruction error between $X$ and $\tilde{X}$ is then calculated and considered in the cost function $J$, defined as follows:

$$J = \frac{1}{2m}\sum_{i=1}^{m}\|\tilde{x}_i - \vec{x}_i\|^2 + \frac{\lambda}{2}\sum_{j=1}^{n_l}\|W_j\|^2 \tag{11}$$

where the first term represents the total reconstruction error across all the $m$ samples and the weight decay term is considered to control the magnitude of the weights for addressing the issue of overfitting. Note that $n_l = 2$ is the number of layers for the traditional shallow autoencoder. The training process of a traditional autoencoder is carried out through iteratively minimizing the cost function $J$ with respect to $W$ and $b$ via back-propagation method [30]. Finally, the training procedure terminates when $J$ converges to a small value.

### B. Shallow autoencoder network for classification

As illustrated above, traditional autoencoder aims to minimize the reconstruction error between the visible layer and the reconstruction layer. However, in this case, the main target of using autoencoder is to automatically extract efficient representative features that can be well classified by a simple classifier. The conventional shallow autoencoder network with supervised fine-tuning is presented in Fig. 3. It involves an additional binary logistic regression classifier connected to the hidden layer. Binary logistic regression aims to estimate the probability that the operating point is unsafe given the extracted features including label of contingencies. Let $y = [y_1, ..., y_m]$ denote the target class vector, we have:

$$y_i = \begin{cases} 0, safe \\ 1, unsafe \end{cases} \tag{12}$$

for $i = 1, ..., m$. Given the extracted features $C = [\vec{c}_1, ..., \vec{c}_m]$, the probability $\hat{p}_i$ of $y_i = 1$ given $\vec{c}_i$ can be estimated as

$$\hat{p}_i = Pr(y_i = 1|\vec{c}_i) = \frac{1}{1 + \exp(-\beta \cdot \vec{c}_i)} \tag{13}$$

where the vector of parameter $\beta$ is typically optimized via some appropriate method (e.g., gradient descent) that aims to find the optimal parameters that a hyper plane can partition the data points into its respective classes with maximum accuracy [31]. In this way, the probability $q_i$ of $y_i = 0$ given $\vec{c}_i$ can be expressed as $\hat{q}_i = Pr(y_i = 0|\vec{c}_i) = 1 - \hat{p}_i$. During the training process, instead of minimizing the cost function $J$ defined in equation (11), the performance of the classification model

with extracted features is measured based on the average of all cross-entropies across $m$ samples, calculated as:

$$L = -\frac{1}{m}\sum_{i=1}^{m}[p_i \log \hat{p}_i + (1-p_i)\log(1-\hat{p}_i)] \quad (14)$$

where $p_i$ represents the true probability of $y_i = 1$. Then back-propagation is employed through the autoencoder network to fine-tune the weights and bias for minimizing $L$. In particular, gradient decent [32] and stochastic gradient decent [33] are two effective methods that can be used for fine-tuning.

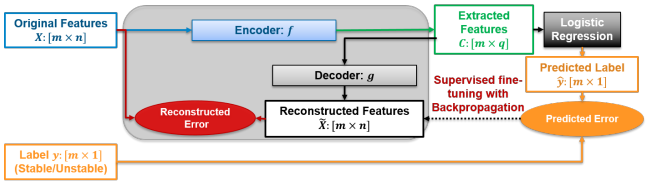



Fig. 3. Shallow autoencoder network with supervised fine-tuning.

### C. The proposed deep autoencoder feature extractor

Based on the plenty of advantages of deep learning illustrated in Section I, the proposed deep autoencoder feature extraction framework can be illustrated with the following contributions and benefits.

i) For system security assessment, traditional feature extraction method (e.g, shallow autoencoder network) has limited performance in processing raw data. Extensive domain expertise and careful engineering are required to design a feature extractor that can transform the raw data into a effective feature space which the classifier could easily classify the security status of the input operating points [16]. As one of the most powerful representation learning techniques, deep learning enables it possible to automatically identify the representative features for classification based on raw data. In this paper, given the original features $X$, each nonlinear module of the deep neural network aims to transform the representation from one level into a higher level with more abstract features. As such, very complex nonlinear relationship between the operating points and their corresponding security status can be learned by carrying out sufficient number of nonlinear transformations through multiple processing layers in a deep neural network. Higher layers of extracted feature make efforts to minimize the uncorrelated variations for classification and strengthen the information of original features that are important for discrimination. The main advantage of using deep network structure is to automatically extract effective features for different tasks rather than manually selected or designed by experts with domain knowledges.

ii) Typically, the class of safe states accounts for an extremely larger proportion than the class of unsafe states, thus introducing the problem of imbalanced data classification. In addition, it is more critical to classify unstable operating points as stable for system operation. For example, if we use "accuracy" to represent the classification error during the training procedure, a 98% accuracy can be easily achieved by misclassifying 2% of unsafe operating points, however, it will result in significant operational issues if 2% points (actual unsafe) are incorrectly predicted as safe. Consequently, a novel objective-based loss function is proposed in this paper to deal with both of the above-mentioned issues.

Fig. 4 illustrates the overall framework for the proposed features extraction method that consists of three main stages: unsupervised pre-training, training, and supervised fine-tuning. As one of the key breakthroughs in deep learning, the authors in [32] have demonstrated that greedy layer-wise pre-training based on restricted Boltzmann machine (RBM) can effectively initialize the weights for deep neural network to ensure the performance of supervised fine-tuning. As illustrated in [34], the success of the layer-wise pre-training strategy is because better initial weights of all layers can mitigate the complex optimization problem of deep networks. In addition, authors in [34] verified that, instead of using RBM, an autoencoder can also be used as a layer construction block when performing the layer-wise greedy unsupervised pre-training.

Let $L$ and $N_L = [n_1, ..., n_L] \in \mathbb{R}^L$ denote the total number of layers and the number of neurons in each layer, then we have $n_1 = n$ and $n_L = q$ where $n$ and $q$ are defined in Section II. Given the raw input features $X = [\vec{x}_1, ..., \vec{x}_m]^T \in \mathbb{R}^{m\times n}$, as shown in Fig. 4, the layer-wise pre-training procedure is composed of training $L$ blocks of autoencoders whose output layer of one block is used as the input layer of the next one. For the $l^{th}$ block, the input and output layers are actually the $l^{th}$ layer $H_l$ and the $(l+1)^{th}$ layer $H_{l+1} = s(W_l H_l + b_l)$ of the constructed deep neural network, respectively. As such, important and high-order correlations between the activities of neurons in $H_l$ are captured by $H_{l+1}$ for $l = 1, ..., L-1$. Note that we have $H_1 = X$, $H_L = C$, and the training procedure of each autoencoder is illustrated in Fig.2. After pre-training multiple layers of the proposed feature extractor, the deep encoder and decoder network is unfolded and initialized with the same weights and bias. Subsequently, the global fine-tuning stage updates the initialized weights and bias with backpropagation. As illustrated in the shallow autoencoder part, binary logistic regression classifier is also employed in the proposed deep feature extractor. However, in order to deal with the challenges of imbalanced data and highlight the importance of misclassified unstable states, an objective-based loss function is proposed based on weighted cross-entropies, estimated F1-score, and estimated precision. Concretely, the proposed loss function can be express as follows:

$$L_{OBJ} = L_W(\Pi_1) - \alpha_1 \hat{F1} - \alpha_2 \hat{Pre} + \frac{\lambda}{2}\sum_{j=1}^{n_l}\|W_j\|^2 \quad (15)$$

where $\frac{\lambda}{2}\sum_{j=1}^{n_l}\|W_j\|^2$ is the weight decay term, $L_W(\Pi_1) = -\frac{1}{m}\sum_{i=1}^{m}[\Pi_1 p_i \log \hat{p}_i + (1-p_i)\log(1-\hat{p}_i)]$ represents a sum of weighted cross entropy that allows to trade off precision and recall by defining the cost $\Pi_1 = \beta \cdot \frac{N_{y=0}}{N_{y=1}}$. Furthermore, $\hat{F1}$ and $\hat{Pre}$ present the approximate $F1\text{-}score$ and $Precision$, respectively, based on the estimated probability $\hat{P} = [\hat{p}_1, ..., \hat{p}_m]$ obtained using equation (13). Concretely, we set $\hat{y}_i = 1$ if $\hat{p}_i > 0.5$ and $\hat{y}_i = 0$ if $\hat{p}_i \leq 0.5$ for $i = 1, ..., m$. Then the classification error between $y$ and $\hat{y}$ can be illustrated using a confusion matrix. The definitions

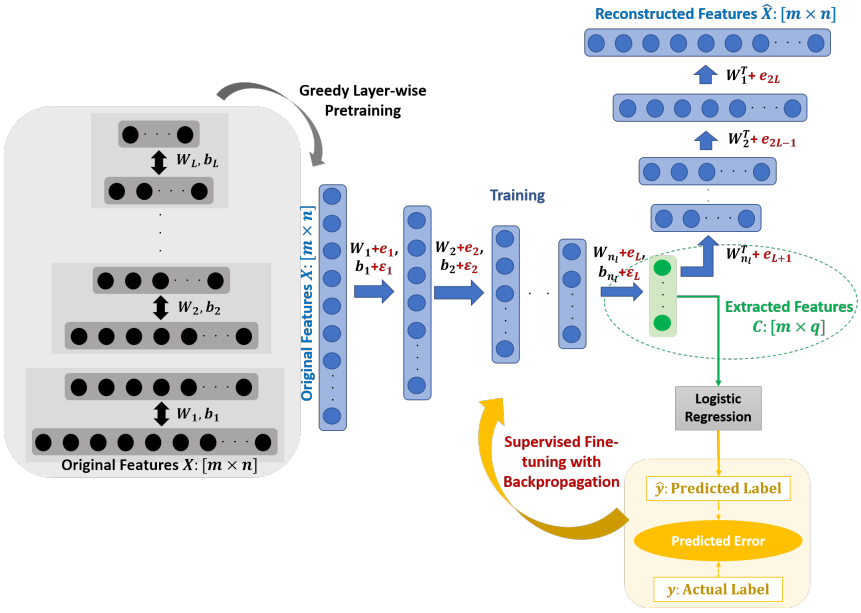



Fig. 4. The proposed deep autoencoder feature extraction framework.

of confusion matrix, $F1-score$, and $Precision$ will be introduced in Section V. Also, $\alpha 1$ and $\alpha 2$ are the user-defined weights for $\hat{F1}$ and $\hat{Pre}$. By using this proposed loss function, the overall classification accuracy can be guaranteed by minimizing the weighted cross entropy term. Meanwhile, the issue of imbalanced data and the importance of incorrectly predicted unsafe states further considered by integrating the estimated $F1$-$score$ and $Precsion$ in the new loss function.

## V. Evaluation Metrics

In this section, four evaluation metrics are proposed to quantitatively evaluate the performance of the proposed method from the perspective of system states classification. Given a machine learning-based classifier (e.g. DT) and a set of training features, the classification quality for a set of test data can be first represented by a confusion matrix, presented in Table I, where TP, TN, FP, and FN denote the numbers of unsafe operating points correctly predicted as unsafe, safe operating points correctly predicted as safe, unsafe operating points incorrectly predicted as safe, and safe operating points incorrectly predicted as unsafe, respectively. In general, there

TABLE I
Confusion Matrix

| | **Predicted Unsafe** | **Predicted Safe** |
|---|---|---|
| **Actual Unsafe** | TP | FP |
| **Actual Safe** | FN | TN |

are varieties of criteria that can be calculated based on these four numbers (e.g. $Accuracy = (TP+TN)/(TP+FP+FN+TN)$). However, for this specific problem, it is more important to focus on the criteria considering the number of operating points which are actually unsafe but predicted as safe (i.e. FP) because this type of classification error may result in significant system security problem. To this end, according to the literature (e.g., [35]), in this paper we employ precision, specificity, and F1-score, defined as follows.

1) In the case of binary security states classification, the precision, also called positive predictive value (PPV), is defined as the proportion of the correctly predicted unsafe states in all the actual unsafe states.

$$precision = TP/(TP+FP). \tag{16}$$

2) The specificity, also known as true positive rate (FPR), represents the proportion of the correctly predicted safe states in all the predicted safe states.

$$specificity = TN/(TN+FP). \tag{17}$$

3) The F1-score is defined as the harmonic mean of the precision and the recall (i.e. $recall(REC) = TP/(TP+FN)$). This metric conveys the balance between the the precision and the recall and reaches its best value at 1 and worst at 0.

$$F1-score = 2 \times PRE \cdot REC/(PRE+REC). \tag{18}$$

4) Accuracy is denoted by the proportion of correct classifications, which can be expressed as follow:

$$accuracy = (TP+TN)/(TP+FP+FN+TN). \tag{19}$$

## VI. Case Study

### A. Test System and Data Description

A modified IEEE 118-bus system [23] consisting of 186 transmission lines, 54 conventional generators, and 10 wind farms of size 100MW each is employed to investigate the proposed framework. The historical dataset, provided by RTE, the French system operator, includes 14250 observations spanning over 7000 nodes containing high-voltage load and wind generation measurements recorded at 5-minute time intervals from January to March 2012. To construct the historical operating points for the test system, 118 load buses and 10

wind generators were randomly selected from the French system and scaled according to the maximum demand values in [36]. The post-fault states are evaluated using DC OPF considering electricity balance constraints, generation operation constraints, and power flow constraints. To evaluate the performance of the proposed method, two case studies were conducted; one with a single outage $C_{single} = [l_{54}]$ and one with four outages $C_{multiple} = [l_{54}, l_{71}, l_{148}, l_{154}]$, where $l_i$ indicates that line #$i$ is in outage.

### B. Methods for Comparison

To evaluate the proposed feature extraction framework compares against other approaches, six methods were tested in total. These methods differ in terms of data and features used when training the final classifier, which in all cases was a DT. In addition, for all methods except M1, which relies on purely historical data, a 10-fold validation scheme was used. The historical database was randomly partitioned in 10 training sets $\{X^k_{train}\}^{10}_{k=1}$ and 10 test sets $\{X^k_{test}\}^{10}_{k=1}$. All methods were implemented using MATLAB and Tensorflow [37] and run on an Intel Xeon PC with 8 cores.

*1) **M1-Historical data**:* The original training features $F_{train} \in \mathbb{R}^{T\times M}$ are directly obtained based on the historical data $X_{train}$.

*2) **M2-Sampled data**:* Instead of using just the historical data, a larger number of samples $\hat{X} \in \mathbb{R}^{T_s\times M}$ are generated via the proposed R-vine copulas model. As such, the size of training features $F_{train}$ is increased from $T$ to $T_s$.

*3) **M3-Sampled data and PCA**:* As one of most broadly used linear feature extraction methods, PCA uses an orthogonal transformation to convert the original features $F_{train}$ into a set of independent principal components (PCs). Let $q$ define the target dimension of the extracted feature, then we can obtain $\hat{F}_{train} = [PC1, ..., PC_q]$ where $q \leq M$.

*4) **M4-Sampled data and SAE**:* M4 uses a single layer shallow autoencoder as the feature extractor to obtain $\hat{F}_{train} \in \mathbb{R}^{T_s\times q}$ for training the security rules.

*5) **M5-Sampled data and DAE**:* A deep autoencoder network with conventional cross entropy loss function.

*6) **M6-Sampled data and DAE+New Loss Function**:* The proposed framework that includes the R-vine copulas sampling strategy, deep autoencoder network, and the objective-based loss function.

### C. Original and Extracted Features

In order to showcase the benefits of the R-vine copulas based sampling strategy and the proposed deep autoencoder based feature extractor, an example of $X^1_{train}$ under the contingency of Line $\#54$ is given in Fig. 5, which presents the extracted features $\hat{F}^1_{train}$ in a two-dimensional space (i.e., $q = 2$) that obtained via different methods. Note that, for Fig. 5 (a) and Fig. 5 (b)-(d), original features $F^1_{train}$ of size $T_{train} \times d = 12,826 \times 604$ and of size $T_s \times d = 100,000 \times 604$ are obtained by carrying out the pre-fault and post-fault simulations based on the historical training dataset $X^1_{train} \in \mathbb{R}^{12,826\times 128}$ and the sampled dataset $\hat{X} \in \mathbb{R}^{100,000\times 128}$, respectively, where $M = 128$ is the dimension of historical stochastic variables. In addition, PCA is performed based on the historical and sampled datasets and then the first two PCs are retained and considered as the extracted features $\hat{F}^1_{train}$ as shown in Fig. 5(a) and Fig. 5(b).

As can be seen, the sampled operating points can successfully enriching the training space with more possible unsafe states, which can potentially enhance the effectiveness of the classification procedure. Nevertheless, the enhanced safe and unsafe spaces are still extremely indistinguishable, which need to be classified by using a very complex non-linear classifier. Fig. 5(c) shows the two-dimensional representation of sampled feature obtained by using a 604-2 shallow autoencoder with supervised fine-tuning. Comparing with Fig. 5(b), a two-dimensional autoencoder exhibits a better visualization regarding the separability of the safe and unsafe points than that of the first two PCs. In particular, most of the unsafe points are mapped to the upper-left space whereas the stable points distributed in the lower-right space with smaller variations than those of PCs. As illustrated in Section III, deep learning can extract higher layers of abstract features that minimize the uncorrelated variations for classification and strengthen the information of original features that are important for discrimination. As shown in Fig. 5(d), a 604-600-500-200-100-2 deep autoencoder with the proposed objective-based loss function clearly outperform the other methods with very separate stable-unstable space and significantly reduced variations when comparing with the other methods. It is also important to note that there exists extremely imbalanced safe (blue) and unsafe (red) states in all sub-figures (i.e., Fig. 5(a)-(d)), thus highlighting the importance of considering the objective-based loss function in the proposed approach. In addition, the extracted features of M6 during the training process for different epochs are shown in Fig. 6. It can be observed that with the increasing number epochs, the distinguishability and variability of the extracted features are gradually improved from $epoch = 1$ to 10, 50 and 100, as shown in Fig. 6.(a), 6.(b), 6.(c), and 6.(d), respectively.

### D. Results for a single transmission line in outage

In this section we demonstrate the effectiveness of the features extracted via different tested methods (i.e., M1-6) for training a system's security rules in the context of a single contingency: line $\#54$. According to a series of tests, M3 has a relatively considerable performance when retaining 100 rather than other numbers of PCs. In order to investigate the effects of the proposed deep structure, the network structure of M4, which is the shallow autoencoder, is set to 604-2, whereas the deep one is set to 604-600-500-200-100-2. Finally, the impact of the proposed objective-based loss function can be investigated by setting the same network configuration parameters for both M5 and M6. For each method the precision, the specificity, and the F1-score of the constructed DTs, averaged across the 10 folds, are given in Table III. Moreover, Fig. 7 presents box plots of the 10 DTs' precision, F1-score, and specificity values for all the tested methods. Also, an example of the number of safe and unsafe states in training and test data for Partition #1 is shown in Table II. It can be observed

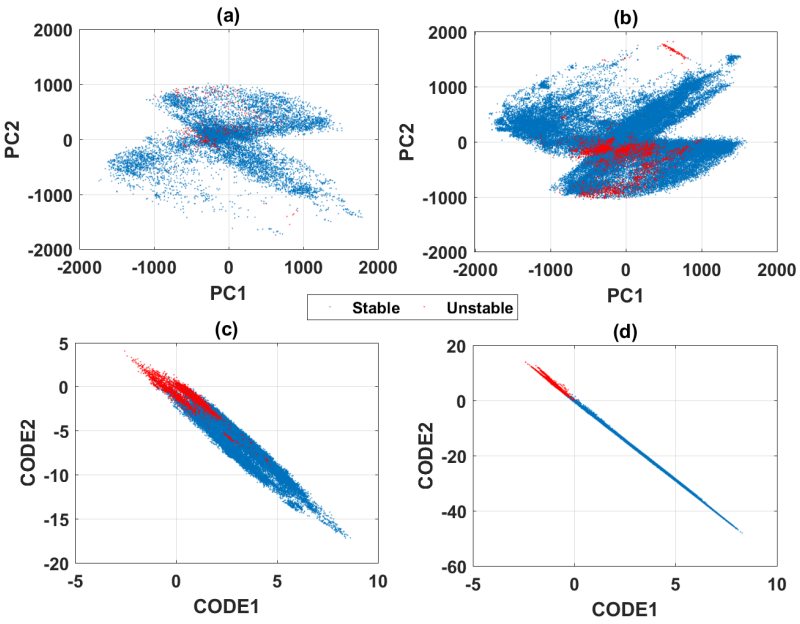



Fig. 5. Scatter plots of two-dimensional representation $\hat{F}^1_{train}$ of 604-dimensional original features $F^1_{train}$ obtained via using the first two PCs of the historical training data (a), using the first two PCs of the sampled training data (b), using a 604-2 shallow autoencoder based on the sampled training data (c), and using a 604-600-500-200-100-2 deep autoencoder based on the sampled training data (d). Red and blue points represent the unsafe and safe states, respectively.

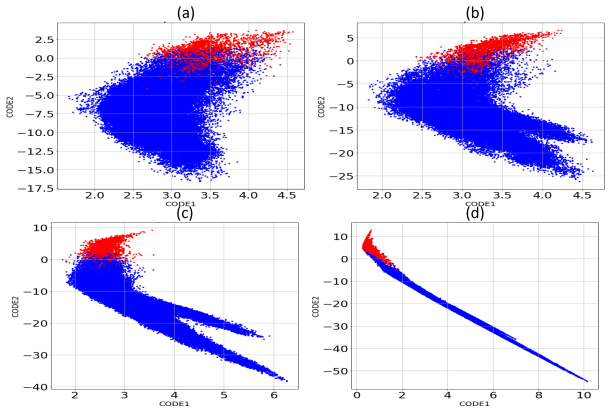



Fig. 6. Scatter plots of two-dimensional representation $\hat{F}^1_{train}$ of 604-dimensional original features $F^1_{train}$ obtained via using a 604-600-500-200-100-2 deep autoencoder (M6) based on the sampled training data: (a) $epoch = 1$; (b) $epoch = 10$; (c) $epoch = 50$; (d) $epoch = 100$.

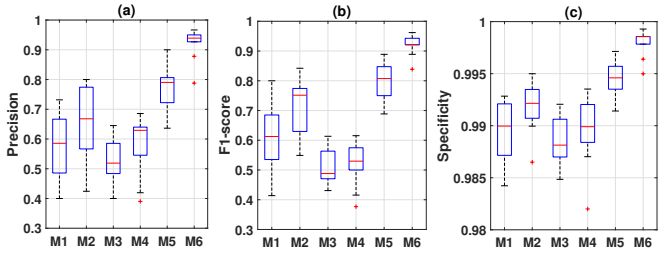



Fig. 7. Precision(a), F1-score(b), and specificity(c) box plots for all the tested methods M1-6. Note that each value corresponds to a partition that constructed for 10-fold validation.

that the number of the unsafe states is significantly less than that of the safe states for both training and test datasets, thus demonstrating the issue of imbalanced classes in system security assessment.

TABLE II
AN EXAMPLE OF NUMBER OF SAFE AND UNSAFE STATES (PARTITION #1)

| | Safe | Unsafe |
|---|---|---|
| **Train (M1: Historical)** | 12,516 | 310 |
| **Train (M2-M6: Sample)** | 97,044 | 2,956 |
| **Test** | 1,392 | 33 |

*1) Historical data vs. Sampled data:* First, the comparison is conducted between M1 and M2 to investigate the benefits of using the enriched training set, as shown in Fig. 5. From Table III, it can be clearly observed that training the decision trees based on the enriched original features can indeed enhance the performance of the derived security rules, which is indicated by the 7.97% increased precision and the 11.05% increased F1-score. Note that the improvement is not significant regarding the metric 'specificity' because of the issue of imbalanced data. As can be seen and Fig. 7, when using the proposed R-vine sampling strategy, 75th percentile of the precision and the F1-score can achieve about 80% and 85%, respectively, which are about 8% and 5% higher than those of M1.

In order to highlight the performance of the proposed feature extractor, the proposed DAE method with new loss function (i.e. M6) is also directly performed based on the historical data without sampling, denoted by M1*, as shown in Table IV. It can be observed that the proposed feature extractor can also significantly improve the classification performance although it is performed based on the limited number of historical data.

TABLE III
AVERAGE EVALUATION METRICS VALUES FOR DIFFERENT SETS OF EXTRACTED FEATURES: SINGLE CONTINGENCY

| | Precision | Specificity | F1-score | Accuracy |
|---|---|---|---|---|
| **M1-Hist** | 57.69% | 98.97% | 61.20% | 95.07% |
| **M2-Sampled** | 65.66% | 99.18% | 72.25% | 97.35% |
| **M3-Sampled+PCA** | 51.51% | 98.85% | 53.97% | 92.94% |
| **M4-Sampled+SAE** | 58.33% | 98.96% | 52.14% | 92.93% |
| **M5-Sampled+DAE** | 77.31% | 99.45% | 80.22% | 98.52% |
| **M6-Sampled+DAEnew** | **92.28%** | **99.81%** | **92.13%** | **99.56%** |

TABLE IV
AVERAGE EVALUATION METRICS VALUES FOR M1 AND M1*

| | Precision | Specificity | F1-score | Accuracy |
|---|---|---|---|---|
| **M1-Hist** | 57.63% | 98.97% | 61.20% | 95.07% |
| **M1*-Hist+DAEnew** | 87.04% | 99.49% | 88.68% | 98.65% |

*2) Unsupervised feature extraction vs. Supervised feature extraction:* Based on the original features obtained using the sampled data, M3 and M4 aim to extract important features in an unsupervised and supervised fashion, respectively. Specifically, M3 employs PCA which makes efforts to retain as much of the variance as possible. However, the target of this work is to extract discriminable features for classification. As a result, comparing with M2, M3 exhibits worse performance (100 PCs were retained) indicating that the rotation direction of maximum information retainment is not coincident to the direction of maximum discrimination. On the other hand, the supervised shallow autoencoder (M4) has better performance than PCA in terms of the approximately 7% improved average precision value. Nevertheless, it is still lower than that of the "no feature extraction" case (i.e, M2). These results highlight the difficulty of the task at hand as well as the fact that feature extraction can actually lead to worse-performing classifiers.

*3) Shallow autoencoder vs. deep autoencoder:* In order to investigate the impact of network depth on the classification performance of the extracted features, a comparison is conducted between the shallow autoencoder network (i.e., M4:604-2) and the deep autoencoder network (i.e., M5:604-600-500-200-100-2). According to the results shown in Table III and Fig. 7, significant improvements are achieved by the 6-layer autoencoder neural network. In particular, the average precision is increased from 58.33% to 77.31% while F1-score also exhibits an approximately 28% growth regarding the mean value of all partitions. Although such significant enhancement can be obtained with the increased network depth, it is imperative to note that it could suffer from the over-fitting problem if the constructed network is too deep. To this end, the enriched training space obtained by using the proposed R-vine sampling strategy may potentially deal with the issue of over-fitting to some extent.

*4) Conventional loss function vs. objective-based loss function:* Finally, the superior performance of deep autoencoder networks with the proposed objective-based loss function can be illustrated by the significantly improved Precision, Specificity, and F1-score as well as their decreased variation across different partitions, as shown in Fig. 7. In particular, the average precision and the average F1-score both approached 92%, which are about 15% and 12% higher than those of M5. These results emphasize that explicitly considering the imbalanced nature of the problem at hand and the severity of the different fault types within the loss function can significantly improve performance.

### *E. Results for all N-1 line outages*

A comprehensive analysis of all 186 N-1 line faults is presented in this part. The heatplots of precision, specificity, F1-score and accuracy obtained via M1 and M6 for each contingency are shown in Fig. 8 and Fig. 9, respectively. Note that the contingencies with values in gray color are cases where all post-fault states are safe (e.g. due to the presence of large parallel lines). As can be seen, the dark red areas indicating low performance metrics in Fig. 8 can be significantly improved to much higher values as shown in Fig. 9, indicated by light color, when using the proposed feature extractor M6. This improvement is particularly pronounced in the contingencies that have poor performance under M1 (e.g. lines 54, 138-145, 158-162 and 177).

Additionally, Table V shows average metric values across all 186 faults for all six methods. As can be seen, all metrics are improved under M6 by, on average, about 3%. Note that since a large number of contingencies already perform well under M1, the performance improvement for the problematic contingencies is much more pronounced. Overall, the observations made when analyzing all N-1 contingencies are similar to the ones obtained when looking at the Line 54 case which was already analyzed in depth in the paper. In particular, M6 achieves the most accurate classification whereas M3 performs the worst. Deep autoenconder (M5) has better performance than the shallow autoenconder (M4), while the proposed loss function modification offers substantial improvement (M6 vs. M5), especially in terms of precision. In addition, the proposed sampling strategy (M2) can effectively enhance the performance by enriching the training database.

In the literature, most of prior work were carried out either based on a relatively small system (e.g., 39-bus system [5]), or using a small set of simulation scenarios (e.g., [38]), or considering "accuracy" as the evaluation metrics (e.g., [5], [38], [39]), which can achieve high values such as over 99%. However, as discussed in our paper, the Accuracy short of full performance characterization in the case of data with imbalanced classes. Therefore, we use other types of metrics to evaluate the performance and it can be observed that the proposed framework is capable of achieving high average Precision and F1-score values over 99%, indicating high quality performance.

TABLE V
AVERAGE EVALUATION METRICS VALUES FOR DIFFERENT SETS OF EXTRACTED FEATURES ACROSS ALL 186 CONTINGENCIES

| | Precision | Specificity | F1-score | Accuracy |
|---|---|---|---|---|
| **M1-Hist** | 96.47% | 99.65% | 96.01% | 99.54% |
| **M2-Sampled** | 97.31% | 99.72% | 97.67% | 99.69% |
| **M3-Sampled+PCA** | 89.53% | 99.51% | 90.02% | 98.73% |
| **M4-Sampled+SAE** | 92.73% | 99.58% | 91.96% | 99.12% |
| **M5-Sampled+DAE** | 97.53% | 99.75% | 98.62% | 99.71% |
| **M6-Sampled+DAEnew** | **99.16%** | **99.84%** | **99.03%** | **99.78%** |

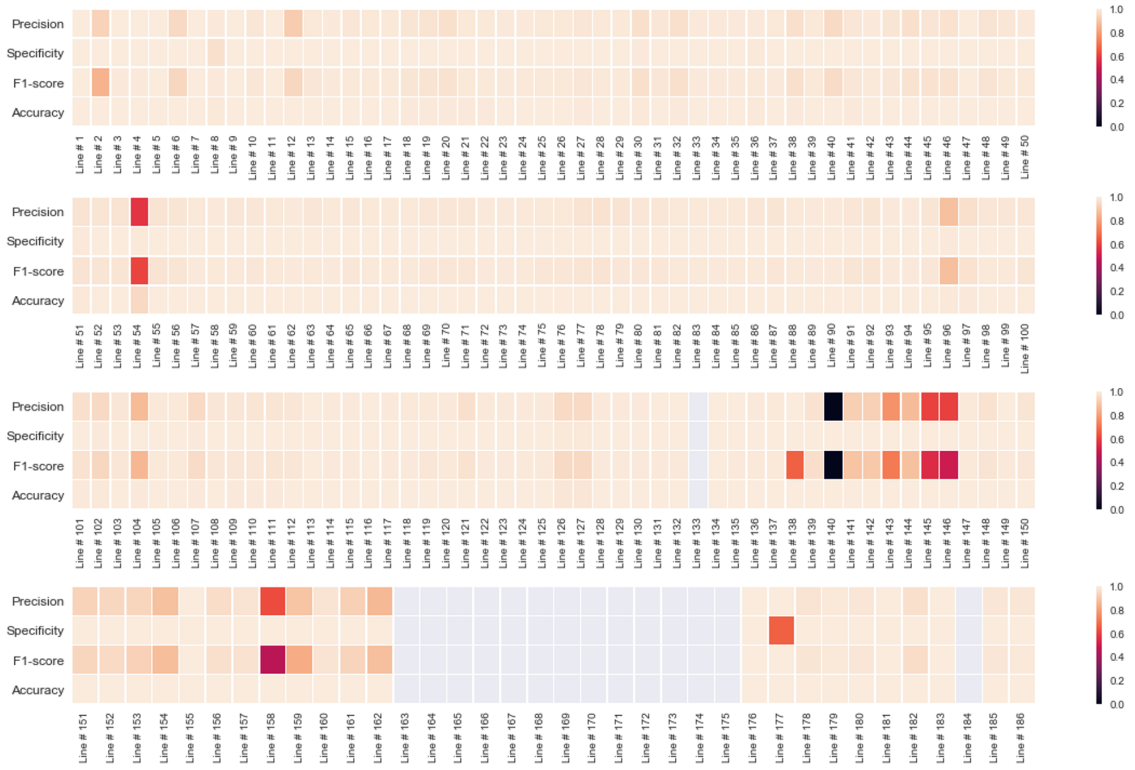



Fig. 8. Heatplots of average precision (a), specificity (b), F1-score (c), and accuracy (d) across ten partitions for all N-1 line faults (i.e., 186 lines) obtained via using M1.

TABLE VI
AVERAGE EVALUATION METRICS VALUES FOR DIFFERENT SETS OF EXTRACTED FEATURES: DIFFERENT CONTINGENCIES (FOUR LINES)

| | Precision | Specificity | F1-score | Accuracy |
|---|---|---|---|---|
| **M1-Hist** | 81.71% | 99.54% | 82.32% | 98.73% |
| **M2-Sampled** | 90.13% | 99.75% | 91.88% | 99.52% |
| **M3-Sampled+PCA** | 70.56% | 99.24% | 69.51% | 96.98% |
| **M4-Sampled+SAE** | 76.46% | 99.42% | 76.61% | 98.12% |
| **M5-Sampled+DAE** | 91.48% | 99.78% | 91.95% | 99.52% |
| **M6-Sampled+DAEnew** | **92.93%** | **99.82%** | **91.98%** | **99.54%** |

### *F. Results for a multi-contingency autoencoder*

In the previous section we analyzed a large number of contingencies, where a different feature extractor has to be constructed for each individual contingency. Training and tuning multiple deep autoencoder networks can be a time-consuming process. To this end, a more efficient way proposed in this paper is constructing a single deep autoencoder network capable of processing a set of contingencies. This can be achieved by training the feature extractor based on the input features including labels of contingencies obtained via one-hot encoding, as illustrated in Section II. Note that for a given contingencies set, the proposed multi-contingency autoencoder can predict the security status for each contingency separately.

Table VI shows the average precision, specificity, and F1-score of the various methods tested when considering a set of contingencies $C_{multiple} = [l_{54}, l_{71}, l_{148}, l_{154}]$. As can be seen, the performance of all tested methods exhibit high classification accuracy. This is because in a multi-contingency setting more information (i.e. labels for different contingencies) are provided for the repeated set of original features (e.g., power flows, phase angles, etc.). Additionally, it is important to note that the multiple contingency scenario considered in

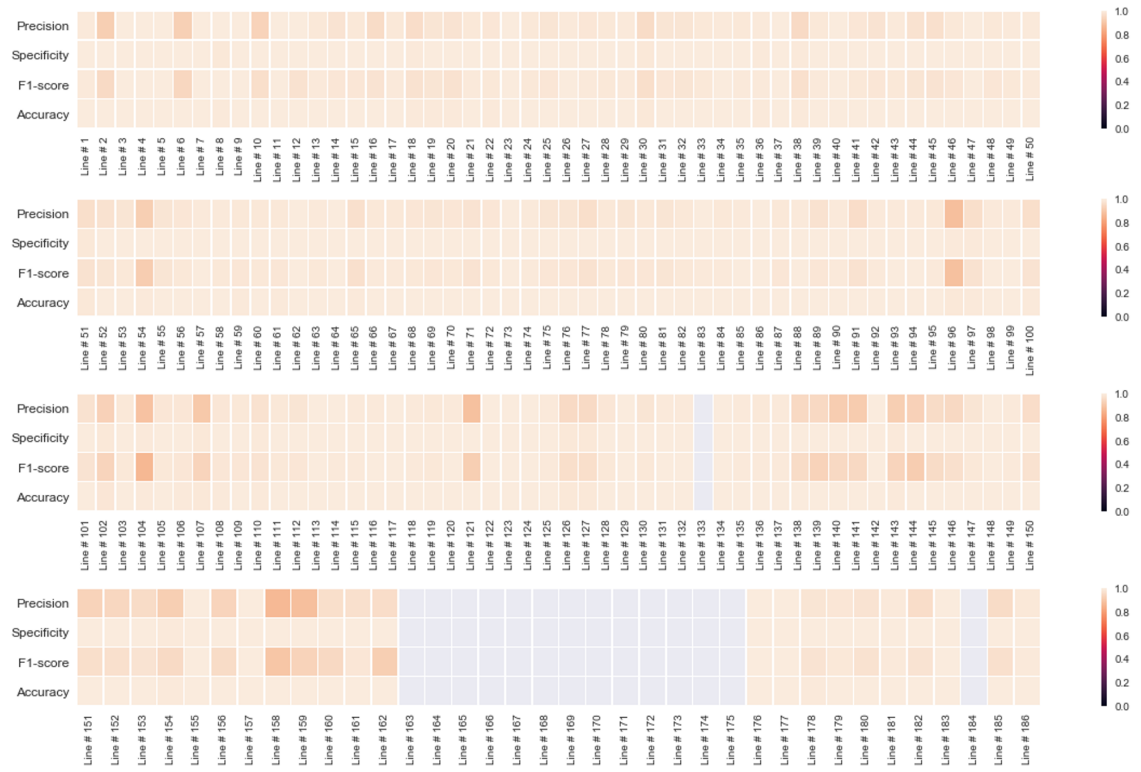



Fig. 9. Heatplots of average precision (a), specificity (b), F1-score (c), and accuracy (d) across ten partitions for all N-1 line faults (i.e., 186 lines) obtained via using M6.

the proposed framework refers to a stack of various N-1 contingencies. Therefore, the complexity of multiple contingency scenario is highly dependent on the complexity of each individual contingency of the considered lines. As shown in Fig. 8, the contingency of Line #54 is a more complicated scenario than Line #71, Line #48, Line #154, represented by lower metric values. As such, classifier performance for the single contingency case of Line #54 is worse than the multi-contingency case because in the latter case, there is an averaging effect across four contingencies (all three of which are easier to predict than Line #54).

As in the previous case, it can be observed that M2, which uses sampled data has better classification performance than M1, which uses solely historical data. In addition, we can see that when using the first 200 PCs extracted via M3, label separability actually decreases. This clearly shows the shortcomings of unsupervised dimension reduction methods for feature extraction. Finally, the benefits obtained by moving from shallow (M4: 604-100) to deep (M5:604-800-700-500-400-350-300-200-150-100) autoencoder, and from a traditional loss function (M5: cross-entropy) to an objective-based loss function (M6: embedded with estimated precision and F1-score), are clearly demonstrated via the progressively improving evaluation metrics. Note that under this multiple contingency scenario, although the performance improvement of the proposed loss function (M6) is smaller than the single contingency scenario shown in Table III, even a small improvement in average performance (e.g., 1%) can mean a very meaningful improvement for individual cases due to the above-mentioned averaging effect across different levels of fault complexity.

The average computational time over ten partitions for all the tested methods are given in Table VI. As can be seen, training the deep autoencoder network (M5) is approximately 1.6 times slower than training the shallow network (M4). In addition, the proposed loss-function (M6) results in slightly longer training times compared to M5. Simpler methods such as M1, M2 and M3 take much less time but suffer greatly in terms of performance.

TABLE VII
AVERAGE CPU TIMES FOR CLASSIFIER TRAINING (SECONDS)

| M1 | M2 | M3 | M4 | M5 | M6 |
|---|---|---|---|---|---|
| 24.04 | 319.53 | 150.83 | 2586.28 | 4210.13 | 4282.90 |

### G. Single vs Multiple

Beyond the case of four-line contingencies, Table VIII shows the average evaluation metric values for all 186 contingencies obtained via the proposed single deep autoencoder network. As can be seen, compared with the results of M6 shown in Table V (i.e., M6-Single), which obtained by training an autoencoder for each contingency and calculating the average metric values over all contingencies, the proposed muti-contingency autoencoder is also shown to be capable to process all possible contingencies in the network with reliable performance. It is imperative to note that there is a trade-off between the classification performance and the computational

time when training a single autoencoder for each contingency or for all the contingencies. In other words, although the performance of the muti-contingency autoencoder (i.e., M6-Multiple) exhibits slightly lower metric values than M6-single, it will significantly reduce the computational complexity to select the appropriate hyper-parameters and training the model for a single deep autoencoder instead of building different autoencoders for different contingencies (e.g., 186 autoencoders).

TABLE VIII
AVERAGE EVALUATION METRICS VALUES: ALL 186 CONTINGENCIES

| | Precision | Specificity | F1-score | Accuracy |
|---|---|---|---|---|
| **M1 (Multiple)** | 96.23% | 99.66% | 96.47% | 99.42% |
| **M6 (Multiple)** | 98.53% | 99.76% | 98.61% | 99.72% |
| **M6 (Single)** | 99.16% | 99.84% | 99.03% | 99.78% |

## VII. CONCLUSIONS

This paper proposes a novel deep learning-based feature extraction framework for system security assessment. The framework consists of an R-vine fitting/sampling module for enriching the available dataset and a feature extractor utilizing deep autoencoders specifically tuned to handle populations with imbalanced classes. The superiority of the proposed approach was demonstrated through a series of case studies involving single and multiple contingencies. In particular we showed that a deep learning-based feature extractor outperforms other approaches in terms of precision, specificity, and F1-score. Furthermore, the benefits of the proposed sampling strategy, the great learning ability of the considered deep autoencoder network, and the efficiency of the objective-based loss function were also shown.

Future work will focus on developing a probabilistic feature extraction framework by exploiting Bayesian deep learning methods and on investigating scaling potential to even larger systems. Furthermore, a feedback connecting evaluation metrics back to the initial phase of feature selection can be considered in the proposed framework to improve model performance. For practical and mission-critical applications, it is important to further investigate how to mitigate possible discrepancies between simulation results and real-world system behavior since these become critical in fully automated work-flows. On the other hand, in order to make the proposed framework adapt to different systems, or the same system but with topology changes, or other system variants, incremental learning techniques can be employed in our future work to solve the computational issue of retraining a new deep network.

## REFERENCES


[1] M. De Jong, G. Papaefthymiou, and P. Palensky, "A framework for incorporation of infeed uncertainty in power system risk-based security assessment," *IEEE Transactions on Power Systems*, 2017.

[2] I. Konstantelos, G. Jamgotchian, S. H. Tindemans, P. Duchesne, S. Cole, C. Merckx, G. Strbac, and P. Panciatici, "Implementation of a massively parallel dynamic security assessment platform for large-scale grids," *IEEE Transactions on Smart Grid*, vol. 8, no. 3, pp. 1417–1426, 2017.

[3] J. Lv, M. Pawlak, and U. D. Annakkage, "Prediction of the transient stability boundary based on nonparametric additive modeling," *IEEE Transactions on Power Systems*, 2017.

[4] P. Kundur, N. J. Balu, and M. G. Lauby, *Power system stability and control*. McGraw-hill New York, 1994, vol. 7.

[5] D. R. Gurusinghe and A. D. Rajapakse, "Post-disturbance transient stability status prediction using synchrophasor measurements," *IEEE Transactions on Power Systems*, vol. 31, no. 5, pp. 3656–3664, 2016.

[6] M. Pai, *Energy function analysis for power system stability*. Springer Science & Business Media, 2012.

[7] J. Sun and K. Lo, "Transient stability real-time prediction for multi-machine power systems by using observation," in *TENCON'93. Proceedings. Computer, Communication, Control and Power Engineering. 1993 IEEE Region 10 Conference on*, vol. 5. IEEE, 1993, pp. 217–221.

[8] D. You, K. Wang, L. Ye, J. Wu, and R. Huang, "Transient stability assessment of power system using support vector machine with generator combinatorial trajectories inputs," *International Journal of Electrical Power & Energy Systems*, vol. 44, no. 1, pp. 318–325, 2013.

[9] F. Hashiesh, H. E. Mostafa, A.-R. Khatib, I. Helal, and M. M. Mansour, "An intelligent wide area synchrophasor based system for predicting and mitigating transient instabilities," *IEEE Transactions on Smart Grid*, vol. 3, no. 2, pp. 645–652, 2012.

[10] T. Guo and J. V. Milanovic, "Probabilistic framework for assessing the accuracy of data mining tool for online prediction of transient stability," *IEEE Transactions on Power Systems*, vol. 29, no. 1, pp. 377–385, 2014.

[11] J. J. Q. Yu, D. J. Hill, A. Y. Lam, J. Gu, and V. O. Li, "Intelligent time-adaptive transient stability assessment system," *IEEE Transactions on Power Systems*, 2017.

[12] C. A. Jensen, M. A. El-Sharkawi, and R. J. Marks, "Power system security assessment using neural networks: feature selection using fisher discrimination," *IEEE Transactions on power systems*, vol. 16, no. 4, pp. 757–763, 2001.

[13] J. B. Souza Filho and P. S. Diniz, "A fixed-point online kernel principal component extraction algorithm," *IEEE Transactions on Signal Processing*, vol. 65, no. 23, pp. 6244–6259, 2017.

[14] N. Lawrence, "Probabilistic non-linear principal component analysis with gaussian process latent variable models," *Journal of machine learning research*, vol. 6, no. Nov, pp. 1783–1816, 2005.

[15] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[16] H. Shi, M. Xu, and R. Li, "Deep learning for household load forecasting–a novel pooling deep rnn," *IEEE Transactions on Smart Grid*, 2017.

[17] H. Wang, H. Yi, J. Peng, G. Wang, Y. Liu, H. Jiang, and W. Liu, "Deterministic and probabilistic forecasting of photovoltaic power based on deep convolutional neural network," *Energy Conversion and Management*, vol. 153, pp. 409–422, 2017.

[18] Y. Wang, Q. Chen, D. Gan, J. Yang, D. S. Kirschen, and C. Kang, "Deep learning-based socio-demographic information identification from smart meter data," *IEEE Transactions on Smart Grid*, 2018.

[19] Y. He, G. J. Mendis, and J. Wei, "Real-time detection of false data injection attacks in smart grid: A deep learning-based intelligent mechanism," *IEEE Transactions on Smart Grid*, 2017.

[20] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *Journal of Big Data*, vol. 2, no. 1, p. 1, 2015.

[21] Y. Wang, Q. Chen, T. Hong, and C. Kang, "Review of smart meter data analytics: Applications, methodologies, and challenges," *IEEE Transactions on Smart Grid*, 2018.

[22] M. Sun, I. Konstantelos, and G. Strbac, "C-vine copula mixture model for clustering of residential electrical load pattern data," *IEEE Transactions on Power Systems*, vol. 32, no. 3, pp. 2382–2393, May 2017.

[23] M. Sun, I. Konstantelos, S. Tindemans, and G. Strbac, "Evaluating composite approaches to modelling high-dimensional stochastic variables in power systems," in *Power Systems Computation Conference (PSCC), 2016*. IEEE, 2016, pp. 1–8.

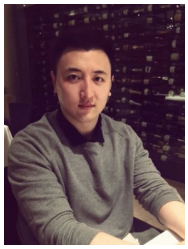
[24] M. Ohsaki, P. Wang, K. Matsuda, S. Katagiri, H. Watanabe, and A. Ralescu, "Confusion-matrix-based kernel logistic regression for imbalanced data classification," *IEEE Transactions on Knowledge and Data Engineering*, 2017.

[25] R. B. Nelsen, "Introduction," in *An Introduction to Copulas*. Springer, 1999, pp. 1–4.

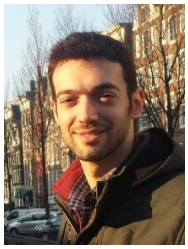
[26] K. Aas, C. Czado, A. Frigessi, and H. Bakken, "Pair-copula constructions of multiple dependence," *Insurance: Mathematics and economics*, vol. 44, no. 2, pp. 182–198, 2009.

[27] E. C. Brechmann, C. Czado, and K. Aas, "Truncated regular vines in high dimensions with application to financial data," *Canadian Journal of Statistics*, vol. 40, no. 1, pp. 68–85, 2012.

[28] J. Dissmann, E. C. Brechmann, C. Czado, and D. Kurowicka, "Selecting and estimating regular vine copulae and application to financial returns," *Computational Statistics & Data Analysis*, vol. 59, pp. 52–69, 2013.
[29] Y. Bengio *et al.*, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.
[30] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.
[31] P. D. Prasad, H. N. Halahalli, J. P. John, and K. K. Majumdar, "Single-trial eeg classification using logistic regression based on ensemble synchronization," *IEEE journal of biomedical and health informatics*, vol. 18, no. 3, pp. 1074–1080, 2014.
[32] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.
[33] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.
[34] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in neural information processing systems*, 2007, pp. 153–160.
[35] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.
[36] Illinois Institute of Technology (IIT), IEEE 118-bus System Data. [Online]. Available: http://motor.ece.iit.edu/Data/IEEE-118%20system%20unit%20data.pdf.
[37] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.
[38] T. Amraee and S. Ranjbar, "Transient instability prediction using decision tree technique," *IEEE Transactions on power systems*, vol. 28, no. 3, pp. 3028–3037, 2013.
[39] F. R. Gomez, A. D. Rajapakse, U. D. Annakkage, and I. T. Fernando, "Support vector machine-based algorithm for post-fault transient stability status prediction using synchronized measurements," *IEEE Transactions on Power Systems*, vol. 26, no. 3, pp. 1474–1483, 2011.



**Mingyang Sun** (M'16) received the Ph.D. degree from the Department of Electrical and Electronic Engineering in Imperial College London, U.K., in 2017. He is currently a Research Associate in Imperial College London. His research focuses on big data analytics in power systems.



**Ioannis Konstantelos** (M'12) received the M.Eng. degree in electrical and electronic engineering from Imperial College London, London, U.K., in 2007, and the Ph.D. degree from the same university in 2013 in the field of electrical energy systems. His research interests include mathematical programming and machine learning techniques applied to the planning and operation of energy systems.

**Goran Strbac** (M'95) is a Professor of electrical energy systems at Imperial College, London, U.K. His current research interests include electricity generation, transmission and distribution operation, planning and pricing, and integration of renewable and distributed generation in electricity systems.