A Confidence-Aware Machine Learning Framework for Dynamic Security Assessment

Zhang, Tingqi ; Sun, Mingyang; Cremer, Jochen L.; Zhang, Ning ; Strbac, Goran; Kang, Chongqing

DOI
10.1109/TPWRS.2021.3059197

Publication date
2021

Document Version
Final published version

Published in
IEEE Transactions on Power Systems

A Confidence-Aware Machine Learning Framework for Dynamic Security Assessment

Tingqi Zhang, *Student Member, IEEE*, Mingyang Sun, *Member, IEEE*, Jochen L. Cremer, *Member, IEEE*, Ning Zhang, *Senior Member, IEEE*, Goran Strbac, *Member, IEEE*, and Chongqing Kang, *Fellow, IEEE*

***Abstract*—Dynamic Security Assessment (DSA) for the future power system is expected to be increasingly complicated with the higher level penetration of renewable energy sources (RES) and the widespread deployment of power electronic devices, which drive new dynamic phenomena. As a result, the increasing complexity and the severe computational bottleneck in real-time operation encourage researchers to exploit machine learning to extract offline security rules for the online assessment. However, traditional machine learning methods lack in providing information on the confidence of their corresponding predictions. A better understanding of confidence of the prediction is of key importance for Transmission System Operators (TSOs) to use and rely on these machine learning methods. Specifically, from the perspective of topological changes, it is often unclear whether the machine learning model can still be used. Hence, being aware of the confidence of the prediction supports the transition to using machine learning in real-time operation. In this paper, we propose a novel Conditional Bayesian Deep Auto-Encoder (CBDAC) based security assessment framework to compute a confidence metric of the prediction. This informs not only the operator to judge whether the prediction can be trusted, but it also allows for judging whether the model needs updating. A case study based on IEEE 68-bus system demonstrates that CBDAC outperforms the state-of-the-art machine learning-based DSA methods and the models that need updating under different topologies can be effectively identified. Furthermore, the case study verifies that effective updating of the models is possible even with very limited data.**

***Index Terms*—Auto-Encoder, bayesian deep learning, confidence awareness, dynamic security assessment, power system operation.**

Manuscript received February 2, 2020; revised June 11, 2020, September 16, 2020, and December 31, 2020; accepted February 6, 2021. Date of publication February 12, 2021; date of current version August 19, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant U20A20159, in part by the National Key Research and Development Program of China 2020YFB1708700, in part by the International (Regional) Joint Research Project of National Natural Science Foundation of China under Grant 52061635101, in part by the CCF-Tencent Open Fund, and in part by the Fundamental Research Funds for the Central Universities (Zhejiang University NGICS Platform). Paper no. TPWRS-00179-2020. *(Corresponding author: Mingyang Sun.)*

Tingqi Zhang and Goran Strbac are with the Department of Electrical & Electronic Engineering, Imperial College London, London SW7 2AZ, U.K. (e-mail: t.zhang17@imperial.ac.uk; g.strbac@imperial.ac.uk).

Mingyang Sun is with the Department of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: mingyangsun@zju.edu.cn).

Jochen L. Cremer is with the Department of Electrical Sustainable Energy, TU Delft, Delft, CD 2628, Netherlands (e-mail: j.cremer16@imperial.ac.uk).

Ning Zhang and Chongqing Kang are with the State Key Lab of Power Systems, Department of Electrical Engineering, Tsinghua University, Beijing 100084, China (e-mail: ningzhang@tsinghua.edu.cn; cqkang@tsinghua.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TPWRS.2021.3059197.

Digital Object Identifier 10.1109/TPWRS.2021.3059197

I. Introduction

THE world is expecting a securer and cleaner power system in the future. To achieve this, continued attention is drawn on the integration of RES. However, due to its intermittent nature, massive uncertainties and corresponding corrective devices are brought to the power system [1]. Suffering from the difficulty of accurately predict the sources (location and levels of power injections), the operation mode of power flows thus is strongly diversified [2]. Consequently, the category of traditional disruptive dynamic phenomena might need to be replaced and expanded with more unforeseen critical conditions, which implies frequent system topology changes. Based on the connectivity status among power system components such as generators, transformers, lines, and loads [3], topology changes can be classified as: 1) scheduled system topology changes (e.g. line maintenance); 2) structure changes caused by the on/off status of circuit breakers. Traditional DSA doctrine hence is challenged by these potential changes of system operation mode, and it is urgently important for TSOs to develop a robust and accurate DSA tool to deal with the challenges of system topology change.

In general, system security issues could be categorized into either static (e.g. line overloading or voltage limits exceeding) or dynamic (e.g. rotor angle stability) security problems. The former is relatively simple since power system parameters in the post-disturbance steady state directly indicate whether system limits are violated or not. The latter, on the other hand, requires more advanced modelling techniques, which can be either data-driven or analytical. In the literature, there are different approaches to predict the transient stability status of a power system: 1) time-domain simulations (TDS), 2) transient-energy-function (TEF) methods, 3) curve-fitting techniques, and 4) machine learning-based methods [4][5]. In particular, TDS provides the most straightforward analytical approach [6]. However, the simulation task is usually highly computing-intensive since detailed information of network configuration during and after a fault is required [4]. In order to solve this issue, researchers have investigated the feasibility of carrying out part of the computation offline. In terms of the TEF methods, Lyapunov function, which includes the kinetic energy and potential energy of a system, is employed to establish a critical energy level first. Then the system assessment is achieved by comparing the target value with this threshold value under a given disturbance [7]. However, one practical issue is that the determination of the level of kinetic and potential energy is almost intractable, especially

under certain disturbances [4]. To tackle this issue, data-driven approaches are developed since they do not require the physical information of the network. Examples such as [8] predict the post-fault rotor angle behaviour using grey Verhulst model. Curve-fitting method (e.g., [9][10]) is another approach which aims at avoiding using the network configuration information. However, the prediction performance is poor as it suffers from the start-up time of prediction and the sampling period [4].

Recently, with the superior development of phasor measurement units (PMUs), the post-disturbance dynamic response of a power system can be directly measured. This cutting-edge technique encourages researchers to construct more reliable models through machine learning methods instead of using conventional rules [11]. In particular, a machine learning model can be established and trained offline by using the TDS results (training labels) collected in advance. The established operating conditions (OCs) provide a region that the system can operate within and likely to occur in the near foreseeable future. Hence the system operator can conduct some analysis before real-time decision-making. As it is released from the constraints of real-time process, classifier thus can be trained on a significantly larger database in order to obtain better performance. In the literature, most of the works are focusing on Decision Trees (DTs) (e.g. [12]–[17]) since it shows advantage on computational speed. Works such as [18] also uses Decision trees to provide interpretability. Other techniques such as Support Vector Machines (SVMs) [19], long short-term memory (LSTM) networks [20], and ensemble approaches [21], [22] have also been widely verified. In addition, works such as [23] and [24] employ hybrid ensemble models, including extreme learning machine (ELM) and random vector functional link networks (RVFL). The former uses the idea of transfer learning in order to implement one model on other faults, so that time cost of training a large number of models can be alleviated. The latter uses generative adversarial network (GAN) in order to complete the missing data so that the original feature characters can be reconstructed. With such, the DSA accuracy can be maintained.

Although machine learning models have shown promising performance in terms of the security assessment task, most of the existing methods are facing the fundamental limitation regarding their capability of confidence awareness. In this work, confidence awareness refers to the ability that a machine learning model obtains model uncertainty through an epistemic learning process. The model with such an ability thus could quantify how confident the model is about its outputs upon the given data set. From the perspective of TSOs, the significance of confidence awareness thus lies in assigning a high level of uncertainty to the erroneous predictions so that the decision-making process could be assisted. Existing works such as [25] and [26] propose probabilistic modelling since it could generate probabilistic intervals. However, these generated intervals cannot be treated as the confidence indicator since the ability of confidence awareness comes from the internal model uncertainty (epistemic uncertainty). This is the property that reflects how much model parameter would change with more knowledge obtained by the model. Therefore, the value of those works is restricted only to proposing more advanced models so that the performance is enhanced.

With the new challenge in confidence representation, in recent years, Bayesian Deep Learning (BDL) has received widespread attention in a range of research fields such as renewable energy forecasting [27], energy price forecasting [28], semantic segmentation [29], and health-care [30] etc. Through the angle of probability theory, BDL reveals the advantages in terms of uncertainty representation, generalization, and prediction reliability, which makes the neural network more explainable [31]. Currently, there are two different directions to realize Bayesian Deep Learning. The work in [32] uses direct inference with Kullback-Leibler (KL) divergence as minimization target. On the other hand, the authors in [33] employ dropout technique as Bayesian approximation, where the aleatoric part of the uncertainties are used as part of the minimization objective under an unsupervised process.

In this paper, a Bayesian deep auto-encoder based methodological framework is proposed, which is able to solve multi-contingency issue and provide confidence information. Key contributions of this paper can be summarized as follows:

1) A confidence-aware machine learning framework for DSA of the large-scale electrical system is proposed. To the best of the authors' knowledge, this is the first paper that achieves confidence awareness by exploiting Bayesian deep learning in the DSA problem.

2) The concept of conditional training is introduced. The proposed framework thus enhances the performance when facing multi-contingency issue within a single model.

3) A confidence-oriented model updating strategy is proposed. The proposed strategy only requires small sample data to update the model.

4) A series of comprehensive case studies are conducted. The superior and robustness performance of the proposed method is demonstrated and compared with other state-of-the-art approaches, which is based on different system topology.

The rest of this paper is organized as follows. Section II introduces the primary challenges in the area of system security assessment. Section III illustrates the overall framework stage by stage and introduces the proposed Conditional Bayesian Deep Auto-Encoder based classifier. Section IV conducts comprehensive numerical experiments to demonstrate the superior performance of the proposed methodology. Section V draws the conclusions and suggests potential future work.

II. Primary Challenges

The penetration of RES not only injects massive uncertainties but also increases the complexity in the context of the power system modelling and operation. Under this circumstance, the primary challenges addressed in this work are summarized as follows:

1) *The Lack of Ability of Confidence Awareness:* Numerous applications of machine learning approaches in the area of power system have been investigated in the last decades. Despite a promising performance in various tasks, one of the fundamental limitations that restrict the practical implementation is that the

existing models do not have the ability of interpretability. More specifically, most of the current methods cannot capture uncertainties and thus fail to express confidence. From the perspective of TSOs, how to make decisions under the uncertainties of the power system? The complicated reality implies that simply developing the modelling technique to improve its accuracy may not be enough in practice. The TSOs need additional information that shows whether or quantifies how the model is confident about its output (i.e. confidence information). To this end, model uncertainty (epistemic uncertainty) reflects the uncertainty in the model parameters, and the model structure becomes vital. For safety-critical applications, it is also of significant importance to capture the confidence information so that the abnormal data points, which are different from training data sets, could be accurately detected. However, the state-of-the-art machine learning methods usually model the uncertainties by rigidly simulating noises and thus can hardly be explained as an epistemic learning process (i.e. uncertainty can be explained away given enough data) [33]. Hence, what we really need is a more advanced model with the real ability of confidence awareness. The influence from massive injected uncertainties thus can be alleviated, and more importantly, TSOs could be offered the flexibility of system operation in the decision-making process.

2) *Multi-Contingency Issue:* The N-1 security criterion provides a preventive standard for system safety operation. From the perspective of machine learning, the significantly enriched database brings the opportunity of training a better model. However, challenges occur as well that more advanced modelling technique is required in order to make full use of the abundant data. In prior works, the authors in [15] treat the multi-contingency issue as independent tasks, and they set up the DTs model for each contingency. However, when deep learning methods are considered to improve the performance further, this strategy will be challenged with a series of critical issues. One of the most serious is that the total workload of the hyperparameters tuning task grows significantly and becomes nearly infeasible, especially in the context of a real-world large scale system. In contrast, the proposed method in [11] employed one-hot coding so that the contingency label can be included within the training data. It is inspired by the theory of multi-task learning, and the model is expected to learn not only correlations among individual contingencies but also the ability to distinguish the difference within a single model. The work thus has the benefits of using one single deep learning model rather than N-1 models in terms of the computational time. However, it actually sacrifices training efficiency since the data dimension is increased. More importantly, the multi-contingency problem is essentially different from the multi-task problem since the data sets in a multi-task problem come from different sources. Therefore, using a single model could cause conflict and thus affect model performance. To this end, developing a method that fully exploits the contingency information will be investigated in this study.

(3) *Model Updating Strategy When Faced With System Topology Changes:* Topology changes, either scheduled behaviours or those accidentally happened such as circuit breaker faults, could cause fundamental changes of system OCs. Those changes, such as line flows, might be very different for different topology, necessitating timely updating. There are various approaches to deal with system topology changes. One is to employ a real-time system topology monitoring scheme as an indicator. The other is to update the model under an experience-based timely basis. However, depending on the size of the system, the corresponding computational cost, which includes but not limited to data from new contingency domain, personnel and time etc., of updating might vary a lot. More importantly, in some cases, the current model can be kept even if the system topology changes. This brings the potential challenge that in practice, in order to obtain the operational flexibility, how could we avoid the unnecessary updating cost? In other words, could the model have self-confidence awareness about its results so that the system operator is able to aware of the proper time? Two further questions that arise and will be addressed in the following chapters thus are: (1) How does the data size contribute to model updating? (2) What would happen if the model performance is already good enough?

III. Framework and Methodology

To tackle the aforementioned challenges, we propose a novel Conditional Bayesian Deep Auto-Encoder (CBDAC) based DSA framework, as shown in Fig. 1. The framework includes the following stages: off-line training, model updating and online assessment. In particular, the input database of training stage is constructed with the input features and their corresponding labels, which are represented by the pre-fault OCs and the post-fault TDS results, respectively. Furthermore, a validation set is established under different system topologies to identify when to update the model indicated by the model uncertainty. After that, the online part can be conducted by feeding in the real-time measurements. Specifically, the detailed step by step explanation of the framework is given as follows:

A. Database Construction Stage:

The first step is to construct the database, which includes pre-fault OCs data and the corresponding post-fault labels indicating whether safe or not. The pre-fault OCs include active and reactive power (either generation $G^{active}_{original} \in \mathbb{R}^{n\times g}, G^{reactive}_{original} \in \mathbb{R}^{n\times g}$ or load $L^{active}_{original} \in \mathbb{R}^{n\times l}, L^{reactive}_{original} \in \mathbb{R}^{n\times l}$), power flows $F^{active}_{original} \in \mathbb{R}^{n\times f}, F^{reactive}_{original} \in \mathbb{R}^{n\times f}$, voltages $V_{original} \in \mathbb{R}^{n\times v}$, and phase angles $\Theta_{original} \in \mathbb{R}^{n\times\theta}$ of each bus. These simulations together construct the m dimension original training features $X_{original} \in \mathbb{R}^{n\times m}$, where n represents the size of entire data set from one topology and $m = 2\times(g+l+f)+v+\theta$. The corresponding post-fault labels, denoted as $Y_{original} \in \mathbb{R}^n$, where for each element y_i

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

are from off-line computed TDS. It is notable here that we use $T_0, T_1, \ldots, T_K$ to represent the data from various system topology. In addition, within one system topology, various contingencies data are generated in order to expand the OCs domain. Given

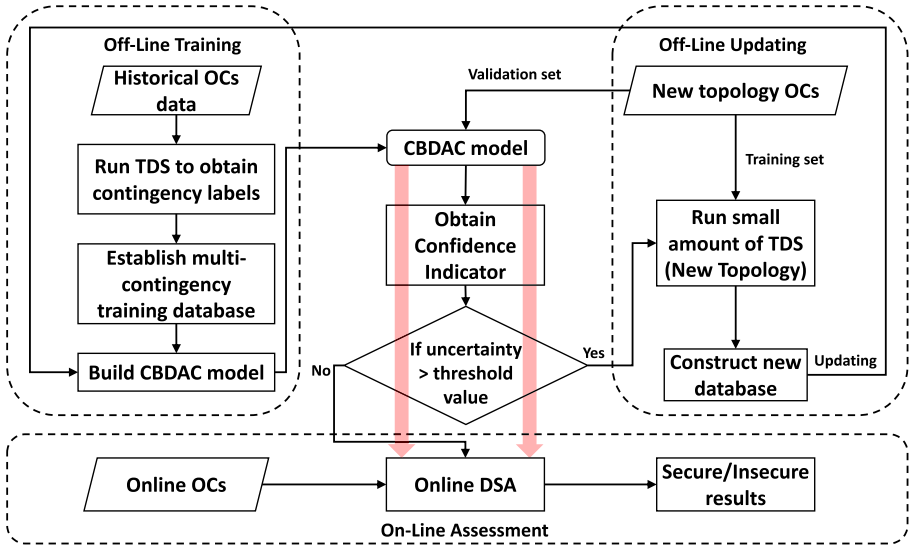

Fig. 1. The proposed CBDAC based DSA framework.

the total contingency number C, data sets $X_{original}, Y_{original}$ are stacked up, normalised and shuffled properly. Eventually, the dimension of features becomes $X \in \mathbb{R}^{(C\times n)\times m}$ and the labels become $Y \in \mathbb{R}^{(C\times n)\times 2}$ since the labels are transferred into one-hot code. The data is then separated into training and testing parts. We use $X^{T_0}_{train}$, $Y^{T_0}_{train}$, $X^{T_0}_{test}$, $Y^{T_0}_{test}$ to denote the data from T_0.

B. Training and Evaluating Stage:

The constructed database $X^{T_0}_{train}$, $Y^{T_0}_{train}$, $X^{T_0}_{test}$, $Y^{T_0}_{test}$ is then used to train and evaluate the model. The model is based on auto-encoder with modified Bayesian approximation, instead of logistic regression, from the hidden layer. In order to enhance the network performance under multiple contingencies, we set up a conditional mask at each layer. A detailed introduction is given in the following subsections.

1) Modified Deep Auto-Encoder: In DSA problem, system OCs are evaluated as secure or insecure, which makes DSA essentially a classification question. To this end, researchers have investigated the feasibility of various machine learning approaches as the classifier. Among those state-of-the-art methodologies, auto-encoder is one of the most famous examples. Traditional auto-encoder is unsupervised, including an input layer, a hidden layer, and an output layer. Auto-encoder is usually used for feature extraction tasks, which minimizes the difference between the input data (coding) and output data sets (decoding). In the area of the power system, auto-encoder based applications are also widely verified such as abnormal state detection [34], system state reconstruction [35], and fault diagnose [36] etc. In [11], a deep auto-encoder with greedy layer-wise pre-training and logistic regression at the hidden layer is demonstrated to have excellent performance in terms of DSA problem. As a further exploration of this work, we continue our work based on this deep auto-encoder structure, which is illustrated in Fig. 2.

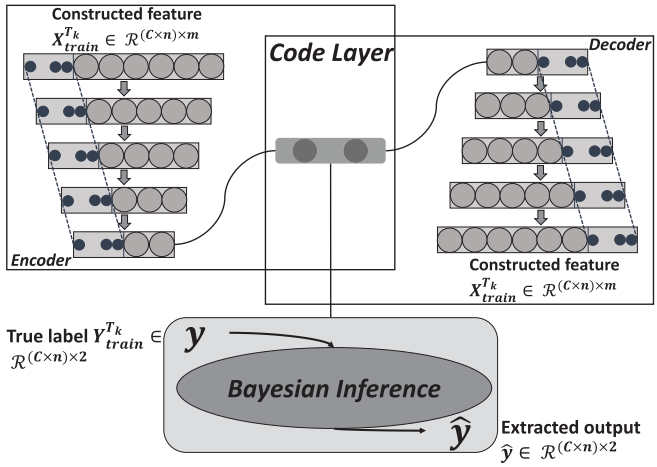

Fig. 2. The structure of conditional bayesian deep auto-encoder.

2) Monte Carlo Dropout as Approximated Bayesian Inference: Although auto-encoder is proved to be effective, as stated in chapter II, the limitation of lacking confidence restricts the practical implementation of this approach in the power system. Hence, we use dropout, a commonly used regularisation technique, to transfer a deterministic auto-encoder into a probabilistic Bayesian model.

A traditional neural network is trained by optimizing its parameters directly. To obtain a probabilistic model, a prior distribution is placed over the weights, usually Gaussian distribution: $\mathcal{N}(0, I)$. With such, a common neural network is transformed into a Bayesian network. How do we train this type of network? We replace the optimising process by minimising KL divergence between the true posterior $p(\omega|X^{T_0}_{train}, Y^{T_0}_{train})$ and the approximating variational distribution $q_\theta(\omega)$, which is usually referred to as variational inference. Due to its intractable

nature, it is extremely difficult to analytically solve this optimization problem. Consequently, this optimization problem is transformed from a KL divergence minimization problem to an Evidence Lower Bound (ELBO) maximization problem. The optimisation function using MC estimator is give by [31]:

$$\hat{\mathcal{L}}_{MC}[\theta] = -\frac{N}{M}\sum_{i\in S}\log p(\hat{y}_i|f^{g(\theta,\epsilon)}(x_i)) + KL(q_\theta(\omega)||p(\omega)) \tag{2}$$

with N, M indicating the sub-sampling process and $g(\theta,\epsilon),\omega$ represents the corresponding parameters.

Although the above optimization approach is straightforward, it has the limitation of high computational burden in terms of the practical implementation [32]. Considering this fact, one simplified way is to use dropout as Bayesian approximation. Dropout can be interpreted as equivalent to variational inference. It generates noise into the feature space, from which we can transform it into the network parameter space as illustrated in equation (3) [31].

$$\begin{aligned}\hat{y} &= \hat{h}\boldsymbol{M_2}\\ &= (h\odot\hat{\epsilon_2})\boldsymbol{M_2}\\ &= (h\cdot diag(\hat{\epsilon_2}))\boldsymbol{M_2}\\ &= h(diag(\hat{\epsilon_2})\boldsymbol{M_2})\\ &= \sigma(\hat{x}\boldsymbol{M_1}+\boldsymbol{B})(diag(\hat{\epsilon_2})\boldsymbol{M_2})\\ &= \sigma((x\odot\hat{\epsilon_1})\boldsymbol{M_1}+\boldsymbol{B})(diag(\hat{\epsilon_2})\boldsymbol{M_2})\\ &= \sigma(x(diag(\hat{\epsilon_1})\boldsymbol{M_1})+\boldsymbol{B})(diag(\hat{\epsilon_2})\boldsymbol{M_2})\end{aligned} \tag{3}$$

In the above equation, we assume an example of a two-layer network with weights $\boldsymbol{M_1}$ and $\boldsymbol{M_2}$ (deterministic matrix), nonlinear activation function σ, and x, h as input into each layer. We use $\hat{x}, \hat{h}$ in order to denote that the input x, h have been through a dropout layer, represented as $\hat{\epsilon_i}$. We write $\hat{W}_1 = diag(\hat{\epsilon_1})\boldsymbol{M_1}$ and $\hat{W}_2 = diag(\hat{\epsilon_2})\boldsymbol{M_2}$, so that it indicates that the network parameters are going through the dropout mask, thus we have:

$$\hat{y} = \sigma(x\hat{W}_1+\boldsymbol{B})\hat{W}_2 = f^{\hat{W}_1,\hat{W}_2,\boldsymbol{B}}(x) \tag{4}$$

The minimisation function of a neural network thus can be rewritten from:

$$\begin{aligned}\mathcal{L}[\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}] &= E^{\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}}(x,y)\\ &\quad+\lambda_1||\boldsymbol{M_1}||^2+\lambda_2||\boldsymbol{M_2}||^2+\lambda_3||\boldsymbol{B}||^2\end{aligned} \tag{5}$$

to:

$$\begin{aligned}\hat{\mathcal{L}}_{dropout}[\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}] &= \frac{1}{M}\sum_{i\in S}E^{\hat{W}_1^i,\hat{W}_2^i,\boldsymbol{B}}(x_i,\hat{y}_i)\\ &\quad+\lambda_1||\boldsymbol{M_1}||^2+\lambda_2||\boldsymbol{M_2}||^2+\lambda_3||\boldsymbol{B}||^2\end{aligned} \tag{6}$$

where i indicates each data point from data sub-sampling with a random set S of size M. According to [37], the first term $E^{\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}}(x,y)$ in neural network optimisation objective function can be rewritten as negative log-likelihood scaled by a constant, as shown in equation (7), where τ indicates the observation noise.

$$\begin{aligned}E^{\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}}(x,y) &= \frac{1}{2}||y-f^{\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}}(x)||^2\\ &= -\frac{1}{\tau}\log p(y|f^{\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}}(x))+constant\end{aligned} \tag{7}$$

Hence we can rewrite equation (6) into (8)

$$\begin{aligned}\hat{\mathcal{L}}_{dropout}[\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}] = &-\frac{1}{M\tau}\sum_{i\in S}\log p(\hat{y}_i|f^{g(\theta,\hat{\epsilon_i})}(x_i))\\ &+\lambda_1||\boldsymbol{M_1}||^2+\lambda_2||\boldsymbol{M_2}||^2+\lambda_3||\boldsymbol{B}||^2\end{aligned} \tag{8}$$

with $\{\hat{W}_1^i,\hat{W}_2^i,\boldsymbol{B}\}=\{diag(\hat{\epsilon_1^i})\boldsymbol{M_1},diag(\hat{\epsilon_2^i})\boldsymbol{M_2},\boldsymbol{B}\}=g(\theta,\hat{\epsilon_i})$ represents the parameters.

Comparing equation (2) and (8), it has been proved that for a specific choice of prior distribution $p(\omega)$, such that:

$$\begin{aligned}&\frac{\partial}{\partial\theta}KL(q_\theta(\omega)||p(\omega))\\ &\quad=\frac{\partial}{\partial\theta}N\tau(\lambda_1||\boldsymbol{M_1}||^2+\lambda_2||\boldsymbol{M_2}||^2+\lambda_3||\boldsymbol{B}||^2)\end{aligned} \tag{9}$$

which is referred to as KL condition, the dropout neural network would have identical optimisation procedure as variational inference [31]. In summary, the minimization function of a dropout approximating network is given as follows [33]:

$$\mathcal{L}[\theta,p] = -\frac{1}{N}\sum_{i=1}^{N}\log p(\hat{y}_i|f^{g(\theta,\hat{\epsilon_i})}(x_i))+\frac{1-p}{2N}||\theta||^2 \tag{10}$$

where N refers to the size of data, p represents the dropout probability and θ is the parameter of the tractable distribution. More details regarding the dropout approximation can be found in the reference [33] and [31].

In terms of the practical implementation, this approximated inference is made by keeping dropout on at both training and testing stage, which is easy to implement. In other words, dropout is done at test stage to sample from the trained network, which can also be treated as a stochastic feed-forward process.

3) Combined Epistemic and Aleatoric Uncertainties in One Model in Classification Tasks: There are two types of uncertainties in Bayesian modelling [38]. The *epistemic uncertainty*, as introduced previously, represents the uncertainty in the model parameters. It is of great significance to capture this type of uncertainty since it could reflect the system topology changes. The *aleatoric uncertainty* on the other hand represents noise inherently in the observations and thus can be further categorized into *homoscedastic uncertainty* and *heteroscedastic uncertainty* [33]. For *homoscedastic uncertainty*, the observation noise parameter σ is fixed while the *heteroscedastic aleatoric uncertainty* varies over different periods of time depending on the data itself. Hence it is obvious that *heteroscedastic aleatoric uncertainty* is a more general and realistic situation. In the prior literature, most of the existing BDL approaches can merely capture *epistemic uncertainty* or *aleatoric uncertainty* alone [31]. Hence, it is important to model these two uncertainties together in one model.

However, it is notable that in our work, the main concentration is on using *epistemic uncertainty* to indicate the proper model updating time. In other words, though discussion of the source of inherent noise (*aleatoric uncertainty*) such as sensor noise, missing data points or any kinds of manually added Gaussian noise etc. are important as well, we assume that noise test and missing data test are out of the scope of this paper and will not be discussed in the following chapters. Due to the page limit, we would like to refer the readers to other valuable works such as [24], [39], [40].

To combine the *epistemic uncertainty* and the *aleatoric uncertainty* in a single model, we need to split the top layers of a deep auto-encoder network into predictive mean $\hat{y}$ as well as predictive variance $\hat{\sigma}^2$, as follows:

$$[\hat{y}, \hat{\sigma}^2] = f_{BDAC}^{\hat{W}}(x) \tag{11}$$

where f_{BDAC} represents the proposed Bayesian deep auto-encoder network. For classification tasks, the output probability is then computed from approximated Monte Carlo integration, which is as follows:

$$p(\hat{y} = c|x, X, Y) \approx \frac{1}{T_{sample}} \sum_{t=1}^{T_{sample}} Softmax(f^{\hat{W}}(x)) \tag{12}$$

The epistemic uncertainty mu of the trained model can then be calculated using the entropy:

$$H(p) = -\sum_{j=1}^{J} p_j \log p_j \tag{13}$$

where J represents the total number of classes.

The overall predictive uncertainty Var$[y]$, consisting of both the *aleatoric uncertainty* and the *epistemic uncertainty* thus can be approximated as

$$\text{Var}[\hat{y}] := H(p) + \frac{1}{T_{sample}} \sum_{t=1}^{T_{sample}} \hat{\sigma}^2. \tag{14}$$

Given that a normal likelihood is chosen to model the aleatoric uncertainty, the final loss function of the BDAC can be formulated as:

$$\mathcal{L}_{classification} = \frac{1}{T_{train}} \sum_{t=1}^{T_{train}} \left(-\hat{y}_{t,j'} + \log \sum_j \exp \hat{y}_{t,j} \right) \tag{15}$$

$$\hat{y}_t = f^{\hat{W}} + \epsilon_t, \epsilon_t \sim N(0, (\hat{\sigma})^2) \tag{16}$$

with $\hat{y}_{t,j}$ the j element in the logits vector $\hat{y}_t$. Note that the loss function can consider both the model uncertainty through $\hat{y}$ and the heteroscedastic uncertainty through $\hat{\sigma}$.

4) Conditional Mask: As illustrated in chapter II, the corresponding expansion of OCs domain caused by the N-1 criterion will result in a significant increase in training burden when using deep learning approaches. Hence, it is reasonable to explore the feasibility of training one model to learn both the unique contingency information and the common operating information. In Fig. 3, we assume a network of L layers where at each

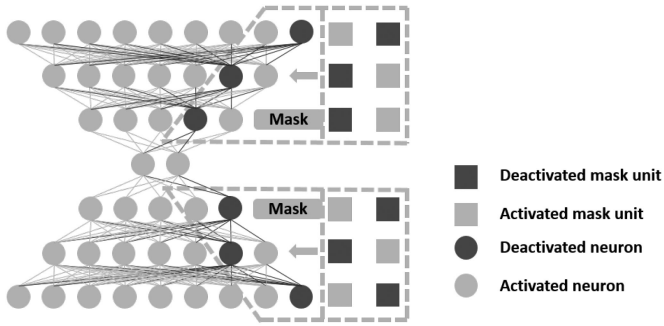

Fig. 3. Auto-encoder with conditional mask.

layer $l \in \{1, \ldots, L\}$, the input and the output of layer are x_l and $\hat{y}_l$ respectively. Given a non-linear activation function $\sigma()$, the feed-forward process within one layer can be written as:

$$\hat{y}_l = \sigma(W_l \cdot x_l + \boldsymbol{B}_l) \tag{17}$$

where W_l and $\boldsymbol{B_l}$ indicate the weights and bias at layer each layer l

We then define the mask layer h, which is consist of one-hot coded 1's and 0's indicating the lines connection or disconnection respectively [41]. Specifically, the width of the mask layer should be equal to the number of contingency. Hence, neurons with mask layer in front are used specifically to learn contingency information. By doing this, the network is separated into two regions. The region without mask will learn and store the common information of various contingencies, while the neurons with mask covered are not activated unless the labeled contingency data comes in. As a result, these neurons with mask do not participate in the training process since they have fixed values and thus zero gradient. Therefore, the model can obtain the ability of dealing with multi-contingency data by exploring the inner structure of the network. The equation for one layer of mask becomes:

$$\hat{y}_{lc} = \begin{cases} h_{lc} \cdot \sigma(W_{lc} \cdot x_{lc} + \boldsymbol{B}_{lc}), & c \in [1, C] \\ \sigma(W_{lc} \cdot x_{lc} + \boldsymbol{B}_{lc}), & c \in [C, D] \end{cases} \tag{18}$$

In the above equation, $\hat{y}_{lc}$ indicates one particular vector output c (column) of layer l. Similarly, W_{lc} and $\boldsymbol{B_{lc}}$ denote the c_{th} vector of the matrix of weights and bias respectively. h_{lc} represents the mask (activation rules) at layer l, colomn c. C denotes the contingency number and D represents the width of the network.

C. Model Updating Stage

When system topology changes, the decision of whether updating the model becomes vital. The practical industrial procedure is to update the model parameters following an experience-based timely basis. Although the current strategy has the advantage of simplicity, it does not participate in or even has a negative contribution to the whole system operation. Hence by employing the property of confidence awareness, the core task of this stage is to provide an indicator of updating, so that redundancy work can be avoided. In addition, when the model is indicated to be updated in a practical situation, TSO is always required to finish

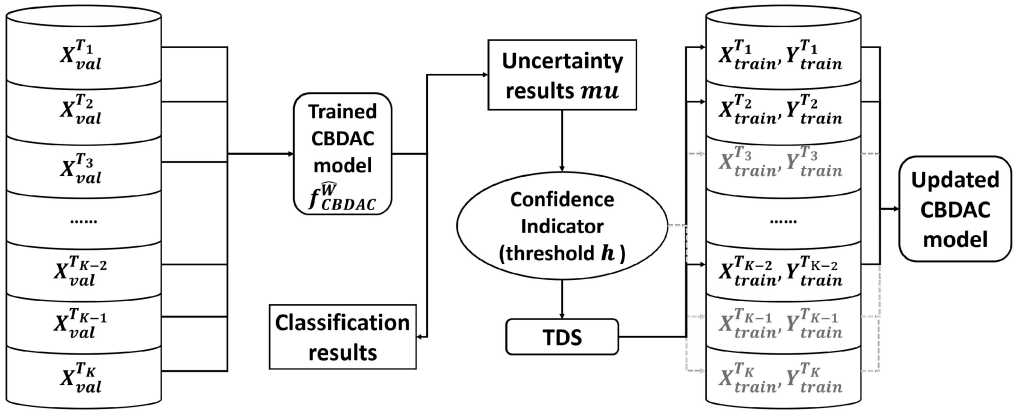

Fig. 4. Model updating stage.

the updating work in limited time. Given that the TDS process of simulating a large training database is a practical bottleneck, the amount of data that required to update model will play a key role.

As can be seen from Fig. 4, validation sets $X_{val}^{T_k}$ from different system topology $T_0, T_1, \ldots, T_K$ are imported into the model firstly. By doing this, an initial classification result $\hat{y_i}$ together with confidence information (model uncertainty) mu, which is defined in equation (13), will be generated and calculated. The confidence information indicates how well the model 'recognizes' the input data. In other words, if the system topology changes, due to the change of data distribution characteristics, mu will increase, which implies that the model does not 'recognize' the data, or feel 'not confident' about its prediction results. It is important to emphasize here that since the TDS are not yet processed for the new topology, the initial uncertainty results can be obtained almost immediately as it only requires stochastic feed-forward calculations based on the CBDAC trained by the original training data. This is of significant importance in terms of rapid, frequent topology changes in the future power system since the reaction time of such an event would be limited. Confidence indicator with a threshold value h is then used to determine whether to update the model. A small number of training labels $Y_{train}^{T_k}$ would be simulated through TDS only when the threshold value h is violated. Therefore, ($X_{train}^{T_k}$, $Y_{train}^{T_k}$) from new topology will be employed in order to update the model, where k represents the number of topologies. Under this circumstance, only essential updating work will be implemented, and the updating task can be guaranteed with reasonable computational burden and accuracy.

To summarize, algorithm 1 demonstrates how our proposed framework works in detail.

Algorithm 1: Confidence-Aware DSA Framework.

Require: $X_{train}^{T_0}$, $Y_{train}^{T_0}$, $X_{val}^{T_k}$, $Y_{val}^{T_k}$, $X_{train}^{T_k}$, $Y_{train}^{T_k}$
Ensure: $X_{train}^{T_0}$, $Y_{train}^{T_0}$, are from topology T_0; $X_{val}^{T_k}$, $Y_{val}^{T_k}$, $X_{train}^{T_k}$, $Y_{train}^{T_k}$, are from topology T_1 to T_K

Define learning rate λ, dropout p, data size N, batch size M, optimizer $Adam$ etc.
Initialize all parameters
function CBDAC$X_{train}^{T_0}$, $Y_{train}^{T_0}$
  repeat
  Mini-batch M optimisation through $\hat{\epsilon} \sim p(\epsilon)$
  Calculate derivation w.r.t. θ
  $\hat{\delta\theta} \leftarrow -\frac{1}{M\tau}\sum_{i\in S}\frac{\partial}{\partial\theta}\log p(\hat{y_i}|f^{g(\theta,\hat{\epsilon_i})}(x_i)) + \frac{\partial}{\partial\theta}N\tau(\lambda_1||\boldsymbol{M_1}||^2 + \lambda_2||\boldsymbol{M_2}||^2 + \lambda_3||\boldsymbol{B}||^2)$
  Update θ: $\theta = \theta + \lambda\delta\theta$
  until θ has been optimised
  return $f_{CBDAC}^{\hat{W}}$
Define threshold value h
Proceed stochastic feed-forward through $f_{CBDAC}^{\hat{W}}$ using $X_{val}^{T_k}$, $Y_{val}^{T_k}$
function Confidence Indicatormu, h
  if $mu > h$ **then**
    Proceed CBDAC($X_{train}^{T_k}$, $Y_{train}^{T_k}$)
  else
    Model can be kept
  return Updated model

IV. Case Study

A. Data Descriptions

The numerical experiments conducted in this study are based on the example 68-bus system [42], which is illustrated in Fig. 5. To simplify our work, we assume that PMUs devices and other measurement devices are deployed to conduct real-time measurement. For instance, voltage magnitudes, phases angles of all buses and the active and reactive outputs of generators can be directly measured and transmitted. Power flow data, on the other hand, can be calculated from the solver. Also, to enrich the OCs domain, more OCs are simulated from a pre-defined range of distribution.

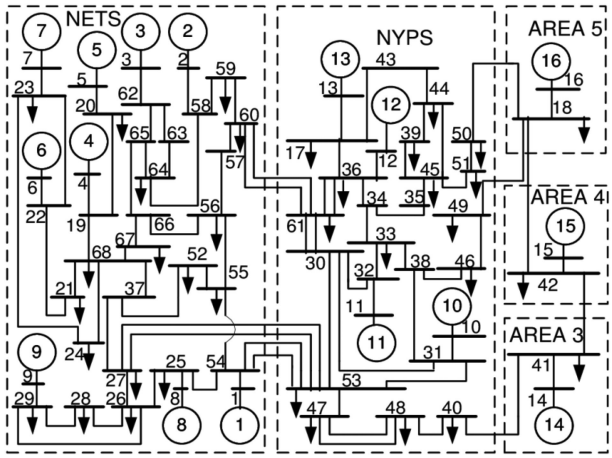

Fig. 5. IEEE-68 bus system [42].

From the system, a set of 12 000 observations are sampled, where each observation represents a pre-fault OC. These observations are created by drawing the active load power from a multivariate Gaussian distribution and using a Pearson's correlation coefficient c between all power pairs. These active load power are then converted to a marginal Kumaraswamy distribution with the probability density function:

$$f(x) = abx^{a-1}(1-x^a)^{b-1} \tag{19}$$

where $a = 1.6$, $b = 2.9$ and $x \in [0, 1]$. The active load power is scaled to be within $\pm 50\%$ of the nominal values, while the reactive load power is scaled by assuming constant impedance of buses. Considering the fact that the resulting OCs might be infeasible and also to restrict the sampled power factor of generators within the range of [0.95, 1], an optimization is solved accounting for the full AC network model. The optimization is carried on in Python 3.5. with Pyomo package and the IPOPT 3.12.4 solver. In order to obtain more general test conditions, the transient stability of three-phase faults is simulated over 22-line contingencies, where the work in [18] has proved the effectiveness of the selected contingencies. These 22-contingencies are selected based on the rules in [43]. For instance, in terms of the fault location, only those close to generator buses are considered, and the fault clearing is coupled with line tripping. The reason is that these are the cases of rotor angle stability interest. As a comparison, cases, where faults are close to loads, are from the angle of voltage stability [44]. The first 14 contingencies are shown in Table III in [43], which are within the NETS part of the IEEE-68 bus system. The rest contingencies are selected based on the same rationale, but for the NYPS part. An OC is considered stable if the differences between each two-phase angles of the generators are within the corresponding limits during 10 s simulation time, otherwise unstable. The fault clearance time is assumed to be 0.1 s. The simulation is performed in Matlab R2016b Simulink, and the model used is described in [42].

TABLE I
HYPERPARAMETERS OF THE PROPOSED CBDAC

Parameter	Value
Layer type	dense
Number of hidden layers	13
Encoder structure	450-350-250-150-50-10
Decoder structure	10-50-150-250-350-450
Logistic regression layer	2
Batch size	15000
Number of epochs	160
Number of samples (T_{sample})	200
Dropout rate	0.001
Optimizer	Adam
Normalisation	[0,1]normalisation
Learning rate	0.0001

TABLE II
COMPUTATION TIME FOR MODEL TRAINING

	CPU Time (s)
DT	31
SVM	18,345
RF	59
DAC	763
BDAC	1,127
CBDAC	831

TABLE III
DIFFERENT TOPOLOGY CASES FOR CBDAC

CBDAC				
Case	**F1-Score**	**ACC**	**PRE**	**SPE**
27-53	92.15%	92.91%	92.52%	93.85%
65-64	91.71%	91.91%	93.71%	94.02%
63-64	91.86%	92.18%	92.03%	92.64%
17-36	64.70%	80.41%	69.30%	88.70%
48-40	79.92%	85.41%	82.66%	90.25%

To establish the database, data from each contingency are finely shuffled firstly in order to provide randomness and generalization. 80% of the entire database (i.e. $80\% \times 12000 \times 22$) is used as the training set to train the model, and the rest 20% of the database is used as the testing set.

B. Experimental Setup

To demonstrate the superior performance of the proposed approach, a series of state-of-the-art methods that have been widely used and firmly demonstrated with reliable performance are used for comparison. For the rest of the paper, the following notation will be used. For instance, DT (*Decision Tree*) SVM (*Support Vector Machine*) RF (*Random Forest*) DAC (*Deep Auto-Encoder Classifier*) BDAC (*Bayesian Deep Auto-Encoder Classifier*) and CBDAC (*Conditional Bayesian Deep Auto-Encoder Classifier*). All the methodologies mentioned above are implemented in Python with the main packages of Scikit-learn [45], Keras [46], TensorFlow [47] and run on an Intel Xeon PC with NIVIDIA Titan-V GPU. The hyper-parameters of the proposed CBDAC model is determined by grid search and cross-validation, which

	ACC	PRE	SPE	F1-Score	CA
DT	81.86%	78.89%	84.02%	78.96%	×
SVM	89.30%	88.77%	91.33%	87.73%	×
RF	89.94%	86.38%	89.97%	88.11%	×
DAC	88.58%	84.51%	88.64%	86.45%	×
BDAC	95.12%	94.65%	95.92%	94.36%	✓
CBDAC	95.80%	96.23%	97.09%	95.19%	✓

Fig. 6. Classification results for different methods.

are given in Table I. The employed evaluation metrics are precision (PRE), specificity (SPE), F1-Score and accuracy (ACC) respectively, where detail introduction is given in the next section.

C. Evaluation Metrics

In this section, the concept of the confusion matrix and four evaluation metrics are introduced to evaluate the performance of security assessment. Given a set of input data, four different types of results can be obtained, which are denoted by True Positive(TP), False Positive(FP), False Negative(FN), and True Negative(TN). For instance, TP represents that unsafe OCs are correctly predicted as unsafe, TN represents when safe OCs are correctly predicted as safe. The wrong results are further grouped into FP, which represents unsafe OCs incorrectly predicted as safe, and FN, when safe OCs are incorrectly predicted as unsafe. The proposed four evaluation metrics are calculated based on these four variables. For instance:

1) The precision: the proportion of the correctly predicted unsafe OCs in all the actual unsafe OCs.

$$precision = TP/(TP + FP) \tag{20}$$

2) The specificity: the proportion of the correctly predicted safe OCs in all the predicted safe OCs.

$$specificity = TN/(TN + FP) \tag{21}$$

3) The F1-Score: the comprehensive evaluation of the the precision and the recall. (Recall = TP / (TP + FN))

$$F1 - Score = 2 \times PRE \times REC/(PRE + REC) \tag{22}$$

4) The accuracy: the proportion of correct classification results over all output results.

$$accuracy = (TP + TN)/(TP + FP + FN + TN) \tag{23}$$

D. Case Study 1: CBDAC Classification Performance Without Topology Changes

In this test, we aim to compare the classification performance of the proposed CBDAC method with other popular methods. As introduced in section IV A, 80% of the database is used to do the training work. Fig. 6 presents the testing results of the four evaluation metrics, where the testing is based on the data size of 20%×12000×22. The length of the blue bar represents the value of the evaluation metric (i.e., a higher value corresponds to a longer bar). CA at the last column indicates the ability of confidence awareness. In terms of the classification performance, the results show that the CBDAC model dominates when compared with the most popular method DT with 17.03%, 21.98%, 15.56%, 20.55% improvements, respectively. In addition, the performance of CBDAC also dominates when comparing with the best of the state-of-the-art methods, RF, especially when considering the fact that the performance is already approaching the limit, with approximately 6.52%, 11.40%, 7.91%, and 8.04% improvements for the four evaluation metrics. Moreover, only Bayesian models have confidence awareness capability. The fact that Bayesian methods provide the best performance indicates the significance of capturing uncertainties.

Regarding the computational time, which is presented in Table II, the experiment shows that DT and RF methods have the shortest training time, 31 and 59 seconds, respectively. In contrast, SVM has the heaviest computational burden, approximately 18 345 seconds. It can be seen that deep learning methods require longer training time than most of the benchmark approaches, where DAC BDAC and CBDAC consume 763, 1127 and 831 seconds, respectively. However, it is notable that model training is an offline procedure. Given the input dataset, the security assessment task can be finished within seconds in practical use. Therefore, the main target in this case is to obtain an accurate classification result.

E. Case Study 2: CBDAC VS. Other Methods Under 44 Different System Topology

In this test, we aim to evaluate the performance of the proposed CBDAC method and other methods when facing system topology changes. The trained models shown in case study 1 are directly used in this test without re-training. OCs data from new topology cases are sampled following the same rationale as introduced in section IV A. We assume system topology changes by switching off lines between buses. We have generated 44 different topology cases with various similarity to the original system topology. For instance, by observing the system structure from [42], we can find that there is a double line scheme between bus NO.27 and NO.53, hence the disconnection between these two buses might have a slight influence on the rest area of the network. On the other hand, bus NO.17 has the maximum load within the system, which implies that disconnection occurred here could cause severe power flow pattern changes. It is notable that due to the extreme time cost, it is unreasonable to keep considering the SVM method.

The general experiments results are shown in Fig. 7, where Bayesian methods show superior performance than the DT, RF and DAC models. For instance, as illustrated in Fig. 7(a), in terms of the comprehensive evaluation F1-Score, DT has the lowest score, an average of 67.80% while RF has a slightly higher performance of 74.45%. Deep Learning methods show significantly better results such as DAC at 77.51% and BDAC at 78.06%. Our improved CBDAC method has the best average performance of 83.31%. CBDAC also outperforms in terms of other metrics, which is demonstrated in Fig. 7(b)–(d) respectively. Table III also shows several individual results of CBDAC model. More importantly, if the acceptable accuracy level is set to be 0.8 F1-Score, we can find that none result from DT method

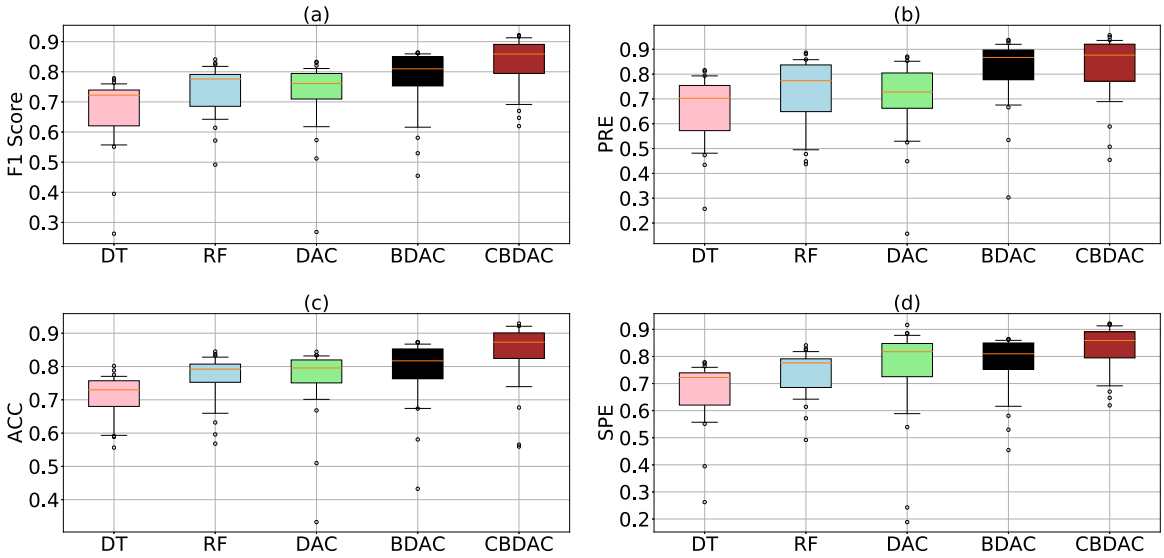

Fig. 7. Box plots of (a) F1-Score, (b) PRE, (c) ACC, and (d) SPE of 44 different topology.

locates above it. RF has a proportion of 20.45% (9 of 44) that goes over the threshold. DAC and BDAC methods have 45.45% (20 of 44) and 54.55% (24 of 44) results that go beyond the threshold value respectively. CBDAC has 70.45% (31 of 44) above the threshold.

The result proves that the Bayesian model with improvement still has superior performance when facing system topology changes, which further implies that resources can be saved if unnecessary updating work can be avoided.

F. Case Study 3: Epistemic Uncertainty as Model Updating Indicator

So far we have proved that the Bayesian deep learning method with improvement is advance and also robust when faced with system topology changes. However, the essence of forecasting (classification) itself determines that we will never know whether the next prediction is true until we know the results. Hence, a reliable auxiliary indicator, which has the ability to show the 'confidence' of the forecasting result becomes vital. Bayesian method thus shows its second advantage: the ability to represent the prediction confidence using model uncertainty as the indicator. The model uncertainty mu, as previously illustrated in equation (13), represents the domain knowledge learned by the model. In other words, the uncertainty of the Bayesian model can reflect the similarity of the original system topology and the new topology, i.e. a big change of topology means a significantly higher uncertainty.

In this case study, BDAC and CBDAC are used since they are Bayesian-based approaches. It is notable that in this work, we only consider the topology changes that are pre-defined by the TSO. As can be seen in Fig. 4, when system topology changes, a small number of OCs are feed into the trained CBDAC model firstly, and the uncertainty results can be generated and collected. During this process, we find that the generated uncertainty values are around a certain level. Compared with the uncertainty level of a finely trained model at the original topology, it is found that the uncertainty level is correlated with the model performance at each topology case. It is also implied that the level of uncertainty is determined by the characteristics of the database (system topology characteristics). In other words, when it comes to another network (e.g. a 108-bus system), there will be another uncertainty level, which could also be decided by the validation data set $X_{val}^{T_k}$ from the corresponding topology cases. Therefore, considering the trade-off between the model accuracy and the updating work burden, we found that 0.8 F1-Score is a reasonable setting.

In this test, if 0.8 F1-Score is used to distinguish safe and unsafe, 20 out of 44 topology cases will be identified as unsafe when using BDAC model. Similarly, in terms of CBDAC model, 13 topology cases have F1-Score lower than 0.8. The performance of the proposed two models under 44 different topology cases and the effectiveness of uncertainty indicator is demonstrated in Fig. 8. However, it is notable that the uncertainty might not be sensitive enough when it comes to each individual cases. Instead, it is more reasonable to evaluate various set of topology cases and calculate the average performance. For instance, though fluctuation could be observed, as a general calculation, the first 20 topology cases have an average uncertainty of 0.171, which is larger than the average of the last 24 cases: 0.149, in the BDAC figure Fig. 8(b). In terms of CBDAC Fig. 8(d), 0.172 VS 0.154 are the average uncertainty values for the first 13 cases and the rest.

Therefore, we can conclude that if 0.8 F1-Score and 0.16 of uncertainty are chosen to be the threshold value, we can observe that most of the topology with poor performance can be detected. This 'detection' property has significant importance since it is generated simultaneously with the classification results. In other words, in reality, the system operator can have confidence information on the model's prediction, which allows him to decide whether to trust or update the model.

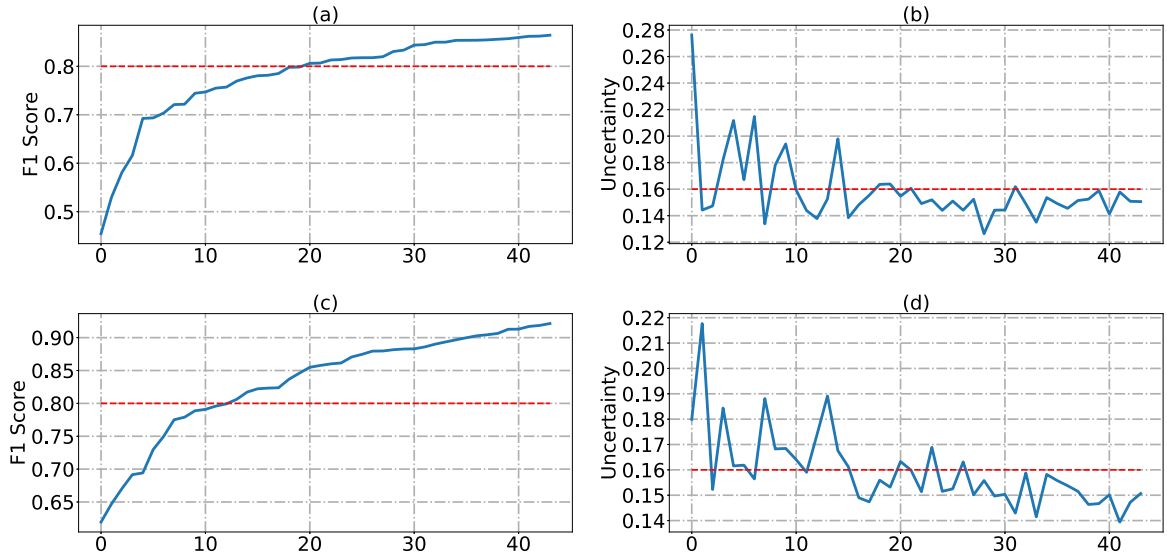

Fig. 8. F1-Score (a)–(c) and Uncertainty (b)–(d) of BDAC (a)–(b) and CBDAC (c)–(d) under 44 different topologies.

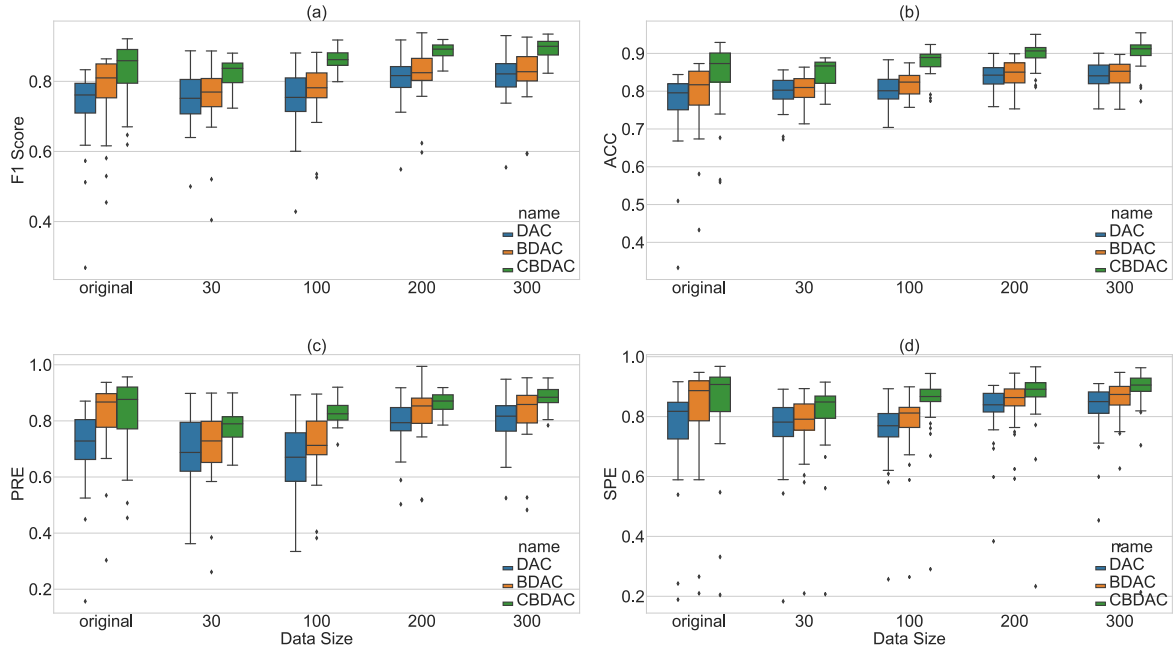

Fig. 9. Model updating with different data size: Box plots show the evaluation of (a) F1-Score, (b) ACC, (c) PRE, and (d) SPE.

G. Case Study 4: Model Updating Using Small Data

In this test, we explore the situation when only limited data is available to update the model. In terms of new topology cases, 800 OCs are randomly sampled for each contingency (i.e. a total of $800 \times 22 = 17\,600$). Following the rationale before, the new database is finely shuffled in order to provide good randomness and generalization. The first 100 OCs of each contingency are used to generate the uncertainty information (i.e. $100 \times 22 = 2200$). The last 400 OCs of each contingency (i.e. $400 \times 22 = 8800$) are used as the testing set. The rest 300 OCs thus are used as the updating data set. Therefore, we design four different updating scenarios, where 30 OCs, 100 OCs, 200 OCs, and 300 OCs are used to update the model, which generates data sample size of 660, 2200, 4400, 6600 respectively. The OCs from each scenario are sampled uniformly except the 300 OCs case so that the test is convincing.

We choose DAC, BDAC, CBDAC in the experiments as a comparison. The evaluation results are shown in Fig. 9. Firstly, we can observe that the classification performance can be improved by updating the model with a few data points. The performance is enhanced as the number of data increases, which is in line with our intuitive speculation. However, one

TABLE IV
PERFORMANCE OF DIFFERENT UPDATING STRATEGIES: 100 OCS

100 OCs Updating				
	F1-Score	**ACC**	**PRE**	**SPE**
Original	83.31%	84.93%	82.93%	84.64%
All	86.30%	87.88%	83.23%	85.03%
Stochastic	84.08%	85.72%	83.10%	84.54%
Selective	86.10%	87.18%	86.03%	86.80%

TABLE V
PERFORMANCE OF DIFFERENT UPDATING STRATEGIES: 30 OCS

30 OCs Updating				
	F1-Score	**ACC**	**PRE**	**SPE**
Original	83.31%	84.93%	82.93%	84.64%
All	82.40%	84.90%	78.11%	81.16%
Stochastic	82.33%	84.31%	80.45%	83.15%
Selective	84.79%	86.08%	84.77%	85.71%

interesting phenomenon here is that when using extremely small size data (e.g. 30 OCs), the performance becomes unstable. For example, the original F1-Score performance of topology NO.10 and NO.11 are 0.9215 and 0.8615. After updating with 30 new OCs, their new performance is 0.8412 and 0.7896 respectively. Considering this phenomenon, it is reasonable to think about a more efficient updating strategy, with the ability to distinguish the necessity of updating the model. The Bayesian model thus is able to fulfil our requirement.

Based on our previous experience from case study 3, it has been found that the uncertainty level is mainly decided by the characteristics of the network. In terms of the example IEEE 68-bus system, we find that 0.16 uncertainty could be a proper separation limit, which identifies 16 cases to be updated. It is notable that this time we only focus on the extremely small size data, i.e. 30 and 100 OCs. As a comparison, we select 16 cases stochastically in order to prove the effectiveness. Tables IV and V demonstrate the average performance of four evaluation metrics in terms of different updating strategies. For instance, traditional updating strategy (i.e. update all) works only if there is sufficient data. However, considering the corresponding time consuming of 2200 data points during the TDS, the practical value of this strategy remains questionable. On the other hand, updating the model selectively is proved to have not only equally robust performance but also significantly fewer time consumption. For example, in our test, only 16 out of 44 topology cases are required to do the TDS, which means the time consumption is only 36% of traditional updating strategy, not to mention the part of re-training the model. Furthermore, comparing to the traditional strategy, the selective updating strategy still has reliable performance, even with significantly fewer data points.

To verify the effectiveness of computational cost reduction of the proposed updating strategy, we illustrate the simple calculation process which is based on several assumptions. Assume the model training time is T for one topology and the TDS time for calculating the label is S for one OC. Comparing to T and S, the model initialization time and the feed-forward time can be neglected. The traditional updating strategy requires the model to be trained every time the system topology changes based on the full-sized database, which means the full-sized database TDS is also needed. In our experiments, the total time consumption thus should be $44 \times (T + 300 \times 22 \times S)$, where 300×22 OCs are used for 22 contingencies. This results in a significantly high computational cost, thus will not be considered. Instead, by using small size database, the computational cost can be reduced to $44 \times (T_{100} + 100 \times 22 \times S)$ or $44 \times (T_{30} + 30 \times 22 \times S)$, where T_{100} and T_{30} indicate the training time using 100 or 30 OCs from each contingency. The training time is also reduced as the data size is reduced, which means $T_{30} < T_{100} < T$. By using the proposed updating strategy, only 16 out of 44 topology need to be updated which further reduces the computational cost to $16 \times (T_{100} + 100 \times 22 \times S)$, or $16 \times (T_{30} + 30 \times 22 \times S)$. Comparing the proposed 'Selective' updating strategy and the 'All' strategy, the approximated computational time saving can be calculated as follows

$$\left(1 - \frac{16 \times (T_{30} + 30 \times 22 \times S)}{44 \times (T_{100} + 100 \times 22 \times S)}\right) \times 100\%$$
$$\approx \left(1 - \frac{16 \times (30 \times 22 \times S)}{44 \times (100 \times 22 \times S)}\right) \times 100\% = 89.09\% \quad (24)$$

where we find in our experiments that $T_{30} \ll 30 \times 22 \times S$, and $T_{100} \ll 100 \times 22 \times S$.

H. Discussion

The proposed Conditional Bayesian Deep Auto-Encoder based DSA classifier has shown promising performance. Specifically, in this work, one practical problem we would like to solve is how to avoid unnecessary cost when system topology changes? We solve the problem by using the confidence as updating indicator and small size data.

Given the complexity of a real power grid, it is usually infeasible for the traditional approaches to scale and adapt to a larger power network. For a medium-sized system such as the IEEE-68, the use of a pre-trained classifier can be much faster than using optimization approach since it solely involves the evaluation of a small number of inequality statements (i.e. security rules). This advantage persists to much larger systems where the computational burden of optimization problems may scale in a non-linear fashion. It can be particularly important when operating in a real-time fashion where the available computational time budget is limited, and the list of contingencies to be checked might include hundreds or thousands of potential faults. However, the most fundamental benefit of the proposed workflow is that it can be readily extended to other types of stability indicators that cannot be determined via optimization but only via TDS (e.g. angle stability, small-signal stability, transient stability etc.). As presented in [16], performing such simulations in real-time is prohibitively slow, which is why an offline analysis must have been carried out beforehand.

The limitation of the work lies in: (1) The database to be used in a ML task is usually collected from various scenarios in advance, and the training works are also done by offline. In other words, in terms of the basic DSA task, there will be

enough time for the TSO to do the database collecting and updating works. In addition, to enrich the OCs domain, more OCs are simulated from a pre-defined range of distribution. These OCs are sampled in order to cover the OCs domain that is potentially to occur in the near future, thus can make the database more effective. However, TSO might only have limited reacting time when there is a topology change. Considering the fact that different topology cases might lead to various reacting time, it is possible that practical updating work could be challenged. (2) The uncertainty threshold value is selected based on experience. Therefore, it is imperative to investigate an analytical method to identify the appropriate threshold for the proposed CBDAC method. (3) More comprehensive evaluation metrics should be proposed or employed to deal with the imbalanced problem of DSA.

V. Conclusion

Machine learning approaches have been proved to be promising in terms of forecasting and classification tasks. Predictably, it will play an important role in the future power system. However, traditional machine learning techniques are lack of capability in confidence awareness, which is of great importance for TSOs to understand whether the data-driven model is certain about its prediction. As a response, this paper proposes a confidence-aware machine learning framework for DSA based on the Conditional Bayesian Deep Auto-Encoder network. The proposed CBDAC model uses dropout to achieve Bayesian approximation with further improvement of conditional training. The superiority and robustness of the proposed methodology are demonstrated with comparison to a series of state-of-the-art methods. We have shown that comparing with the best of state-of-the-art methods, our proposed model still has 6.52%, 11.40%, 7.91% and 8.03% improvement in terms of four evaluation metrics. Furthermore, we explore the feasibility of using limited data in order to update the model and thus propose a selective updating strategy. Indicated by the model confidence, the proposed strategy significantly alleviates the unnecessary time consumption, approximately 89%, under frequent system topology changes, which is of great practical value.

In the future, one potential direction could be how to optimize the updating strategy. Algorithms such as active sampling or incremental learning could be employed and improved. In addition, other works such as missing data or noise testing could be of great practical value. It is also believed that it is of high interest to consider the full workflow from measurements, over data processing and state estimation together.

References

[1] M. Sun, F. Teng, X. Zhang, G. Strbac, and D. Pudjianto, "Data-driven representative day selection for investment decisions: A cost-oriented approach," *IEEE Trans. Power Syst.*, vol. 34, no. 4, pp. 2925–2936, Jul. 2019.

[2] P. Panciatici, G. Bareux, and L. Wehenkel, "Operating in the fog: Security management under uncertainty," *IEEE Power Energy Mag.*, vol. 10, no. 5, pp. 40–49, Sep. 2012.

[3] M. Kezunovic, "Monitoring of power system topology in real-time," in *Proc. 39th Annu. Hawaii Int. Conf. Syst. Sci.*, vol. 10, Jan. 2006, pp. 244b–244b.

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

[5] I. Konstantelos, M. Sun, S. H. Tindemans, S. Issad, P. Panciatici, and G. Strbac, "Using vine copulas to generate representative system states for machine learning," *IEEE Trans. Power Syst.*, vol. 34, no. 1, pp. 225–235, Jan. 2019.

[6] P. Kundur, N. J. Balu, and M. G. Lauby, *Power System Stability and Control*. New York, NY, USA: McGraw-Hill, vol. 7, 1994.

[7] M. Pai, *Energy Function Analysis for Power System Stability* (ser. Power Electronics and Power Systems), USA: Springer, 2012. [Online]. Available: https://books.google.co.uk/books?id=1HDgBwAAQBAJ

[8] H. Deng, J. Zhao, Y. Liu, and X. Wu, "A real-time generator-angle prediction method based on the modified grey verhulst model," in *Proc. 4th Int. Conf. Electric Utility Deregulation Restruct. Power Technol.*, Jul. 2011, pp. 179–184.

[9] M. H. Haque and A. H. M. A. Rahim, "Determination of first swing stability limit of multimachine power systems through Taylor series expansions," *IEE Proc. C. - Gener., Transmiss. Distrib.*, vol. 136, no. 6, pp. 373–380, Nov. 1989.

[10] J. H. Sun and K. L. Lo, "Transient stability real-time prediction for multi-machine power systems by using observation," in *Proc. TENCON '93. IEEE Region 10 Int. Conf. Comput., Commun. Automat.*, Oct. 1993, pp. 217–221.

[11] M. Sun, I. Konstantelos, and G. Strbac, "A deep learning-based feature extraction framework for system security assessment," *IEEE Trans. Smart Grid*, vol. 10, no. 5, pp. 5007–5020, Sep. 2019.

[12] K. Sun, S. Likhate, V. Vittal, V. S. Kolluri, and S. Mandal, "An online dynamic security assessment scheme using phasor measurements and decision trees," *IEEE Trans. Power Syst.*, vol. 22, no. 4, pp. 1935–1943, Nov. 2007.

[13] V. Krishnan, J. D. McCalley, S. Henry, and S. Issad, "Efficient database generation for decision tree based power system security assessment," *IEEE Trans. Power Syst.*, vol. 26, no. 4, pp. 2319–2327, Nov. 2011.

[14] C. Liu *et al.*, "A systematic approach for dynamic security assessment and the corresponding preventive control scheme based on decision trees," *IEEE Trans. Power Syst.*, vol. 29, no. 2, pp. 717–730, Mar. 2014.

[15] L. A. Wehenkel, *Automatic Learning Techniques in Power Systems*. USA: Kluwer Academic Publishers, 1998.

[16] I. Konstantelos *et al.*, "Implementation of a massively parallel dynamic security assessment platform for large-scale grids," *IEEE Trans. Smart Grid*, vol. 8, no. 3, pp. 1417–1426, May 2017.

[17] M. H. Vasconcelos *et al.*, "Online security assessment with load and renewable generation uncertainty: The itesla project approach," in *Proc. Int. Conf. Probabilistic Methods Appl. Power Syst.*, Oct. 2016, pp. 1–8.

[18] J. L. Cremer, I. Konstantelos, and G. Strbac, "From optimization-based machine learning to interpretable security rules for operation," *IEEE Trans. Power Syst.*, vol. 34, no. 5, pp. 3826–3836, Sep. 2019.

[19] 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," *Int. J. Elect. Power Energy Syst.*, vol. 44, no. 1, pp. 318–325, 2013.

[20] J. J. Q. Yu, D. J. Hill, A. Y. S. Lam, J. Gu, and V. O. K. Li, "Intelligent time-adaptive transient stability assessment system," *IEEE Trans. Power Syst.*, vol. 33, no. 1, pp. 1049–1058, Jan. 2018.

[21] C. Liu, F. Tang, and C. Leth Bak, "An accurate online dynamic security assessment scheme based on random forest," *Energies*, vol. 11, no. 7, Jul. 2018, Art. no. 1914.

[22] S. R. Samantara, I. Kamwa, and G. Joos, "Ensemble decision trees for phasor measurement unit-based wide-area security assessment in the operations time frame," *IET Gener., Transmiss. Distrib.*, vol. 4, no. 12, pp. 1334–1348, Dec. 2010.

[23] C. Ren and Y. Xu, "Transfer learning-based power system online dynamic security assessment: Using one model to assess many unlearned faults," *IEEE Trans. Power Syst.*, vol. 35, no. 1, pp. 821–824, Jan. 2020.

[24] C. Ren and Y. Xu, "A fully data-driven method based on generative adversarial networks for power system dynamic security assessment with missing data," *IEEE Trans. Power Syst.*, vol. 34, no. 6, pp. 5044–5052, Nov. 2019.

[25] N. I. A. Wahab, A. Mohamed, and A. Hussain, "Fast transient stability assessment of large power system using probabilistic neural network with feature reduction techniques," *Expert Syst. Appl.*, vol. 38, no. 9, pp. 11112–11119, 2011.

[26] P. N. Papadopoulos and J. V. Milanović, "Probabilistic framework for transient stability assessment of power systems with high penetration of renewable generation," *IEEE Trans. Power Syst.*, vol. 32, no. 4, pp. 3078–3088, Jul. 2017.
[27] H. Lee and B. Lee, "Confidence-aware deep learning forecasting system for daily solar irradiance," *IET Renewable Power Gener.*, vol. 13, no. 10, pp. 1681–1689, 2019.
[28] A. Brusaferri, M. Matteucci, P. Portolani, and A. Vitali, "Bayesian deep learning based method for probabilistic forecast of day-ahead electricity prices," *Appl. Energy*, vol. 250, pp. 1158–1175, Sep. 2019.
[29] J. Mukhoti and Y. Gal, "Evaluating Bayesian deep learning methods for semantic segmentation," *CoRR*, vol. abs/1811.12709, 2018. [Online]. Available: http://arxiv.org/abs/1811.12709
[30] R. Harper and J. Southern, "A Bayesian deep learning framework for end-to-end prediction of emotion from heartbeat," *CoRR*, 2019, *arXiv:1902.03043*.
[31] Y. Gal, "Uncertainty in deep learning," Ph.D. dissertation, Dept. Eng., Univ. Cambridge, 2016.
[32] M. Sun, T. Zhang, Y. Wang, G. Strbac, and C. Kang, "Using Bayesian deep learning to capture uncertainty for residential net load forecasting," *IEEE Trans. Power Syst.*, vol. 35, no. 1, pp. 188–201, Jan. 2020.
[33] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5574–5584.
[34] M. Martinelli, E. Tronci, G. Dipoppa, and C. Balducelli, "Electric power system anomaly detection using neural networks," in *Proc. Int. Conf. Knowl.-Based Intell. Inf. Eng. Syst.*, New York, NY, USA: Springer, 2004, pp. 1242–1248.
[35] Y. Lin, J. Wang, and M. Cui, "Reconstruction of power system measurements based on enhanced denoising autoencoder," in *Proc. IEEE Power Energy Soc. Gen. Meeting (PESGM)*, Aug. 2019, pp. 1–5.
[36] Y. Wang, M. Liu, Z. Bao, and S. Zhang, "Stacked sparse autoencoder with PCA and SVM for data-based line trip fault diagnosis in power systems," *Neural Comput. Appl.*, vol. 31, no. 10, pp. 6719–6731, 2019.
[37] N. Tishby, E. Levin, and S. A. Solla, "Consistent inference of probabilities in layered networks: Predictions and generalization," in *Proc. Int. Joint Conf. Neural Netw.*, 1989, pp. 403–409.
[38] A. D. Kiureghian and O. Ditlevsen, "Aleatory or epistemic? Does it matter?" *Struct. Saf.*, vol. 31, no. 2, pp. 105–112, 2009.
[39] Y. Zhang, Y. Xu, and Z. Y. Dong, "Robust classification model for pmu-based on-line power system dsa with missing data," *IET Gener., Transmiss. Distrib.*, vol. 11, no. 18, pp. 4484–4491, 2017.
[40] 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.
[41] B. Donnot, I. Guyon, A. Marot, M. Schoenauer, and P. Panciatici, "Fast power system security analysis with guided dropout," in *Proc. 26th Eur. Symp. Artif. Neural Netw. (ESANN)*, Bruges, Belgium, Apr. 2018. [Online]. Available: http://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2018-94.pdf
[42] B. Pal and B. Chaudhuri, *Robust Control in Power Systems*. New York, NY, USA: Springer-Verlag, 2005.
[43] G. Anagnostou and B. C. Pal, "Impact of overexcitation limiters on the power system stability margin under stressed conditions," *IEEE Trans. Power Syst.*, vol. 31, no. 3, pp. 2327–2337, May 2016.
[44] C. Balu and D. Maratukulam, *Power System Voltage Stability*. New York, NY, USA: McGraw Hill, 1994.
[45] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
[46] F. Chollet *et al.*, *Deep Learning With Python*. New York, NY, USA: Manning, vol. 361, 2018.
[47] M. Abadi *et al.*, "Tensorflow: A system for largescale machine learning," in *Proc. 12th fUSENIXg Symp. Oper. Syst. Des. Imp.*, (fOSDIg 16), 2016, pp. 265–283.

Tingqi Zhang (Student Member, IEEE) received the bachelor's degree in electrical engineering with renewable energy from The University of Edinburgh, Edinburgh, U.K., and the master's degree in energy and sustainability with electrical power engineering from the University of Southampton, Southampton, U.K. He is currently working toward the Ph.D. degree with Control and Power Group, Imperial College London, London, U.K. His research focuses on big data technique in energy power system.

Mingyang Sun (Member, IEEE) received the Ph.D. degree from Imperial College London, London, U.K., in 2017. He is currently a Tenure-Track Professor under the Hundred Talents Program with Zhejiang University, Hangzhou, China. He is also a Visiting Researcher with Imperial College London. His research interests include artificial intelligence in energy systems and cyber-physical energy system security and control.

Jochen L. Cremer (Member, IEEE) received the B.Sc. degree in mechanical engineering, the B.Sc. degree in electrical engineering, and the M.Sc. degree in chemical engineering from the RWTH Aachen University, Aachen, Germany, in 2014, 2016, and 2016, respectively, and the Ph.D. degree from Imperial College London, London, U.K., in 2020. He is currently an Assistant Professor with the Technische Universiteit Delft, Delft, The Netherland. His research interests include machine learning and mathematical programming applied to the operation and planning of power systems.

Ning Zhang (Senior Member, IEEE) received the B.S. and Ph.D. degrees from the Electrical Engineering Department, Tsinghua University, Beijing, China, in 2007 and 2012, respectively. He is currently an Associate Professor with Tsinghua University. His research interests include multiple energy systems integration, stochastic analysis, simulation of renewable energy, power system planning, and scheduling with renewable energy.

Goran Strbac (Member, IEEE) is currently a Professor of electrical energy systems with Imperial College London, London, U.K. His research interests include electricity system operation, investment and pricing, and integration of renewable generation and distributed energy resources.

Chongqing Kang (Fellow, IEEE) received the Ph.D. degree in 1997 from the Department of Electrical Engineering, Tsinghua University, Beijing, China, where he is currently a Professor. His research interests include power system planning, power system operation, renewable energy, low-carbon electricity technology, and load forecasting.