Robust Online Dynamic Security Assessment Using Adaptive Ensemble Decision-Tree Learning

Miao He, *Student Member, IEEE*, Junshan Zhang, *Fellow, IEEE*, and Vijay Vittal, *Fellow, IEEE*

***Abstract*—Online dynamic security assessment (DSA) is examined in a data-mining framework by taking into account the operating condition (OC) variations and possible topology changes of power systems during the operating horizon. Specifically, a robust scheme is proposed based on adaptive ensemble decision tree (DT) learning. In offline training, a boosting algorithm is employed to build a classification model as a weighted voting of multiple unpruned small-height DTs. Then, the small-height DTs are periodically updated by incorporating new training cases that account for OC variations or the possible changes of system topology; the voting weights of the small-height DTs are also updated accordingly. In online DSA, the updated classification model is used to map the real-time measurements of the present OC to security classification decisions. The proposed scheme is first illustrated on the IEEE 39-bus test system, and then applied to a regional grid of the Western Electricity Coordinating Council (WECC) system. The results of case studies, using a variety of realized OCs, illustrate the effectiveness of the proposed scheme in dealing with OC variation and system topology change.**

***Index Terms*—Boosting, data mining, decision tree, ensemble learning, online dynamic security assessment, transient stability.**

I. Introduction

DYNAMIC security assessment [1] can provide system operators important information regarding the transient performance of power systems under various possible contingencies. By using the real-time or near real-time measurements collected by phasor measurement units (PMUs), *online* dynamic security assessment (DSA) can produce more accurate security classification decisions for the present operating condition (OC) or imminent OCs. However, online DSA still constitutes a challenging task due to the computational complexity incurred by the combinatorial nature of $N-k$ $(k=1,2,\ldots)$ contingencies and the massive scale of practical power systems, which makes it intractable to perform power flow analysis and time domain simulations for all contingencies in real-time.

The advent of data mining techniques provides a promising solution to handle these challenges. Cost-effective DSA schemes have been proposed by leveraging the power of data mining tools in classification, with the basic idea as follows. First, a knowledge base is prepared through comprehensive offline studies, in which a number of predicted OCs are used by DSA software packages to create a collection of training cases. Then, the knowledge base is used to train classification models that characterize the decision rules to assess system stability. Finally, the decision rules are used to map the real-time PMU measurements of pre-fault attributes to the security classification decisions of the present OC for online DSA. The data mining tools that have proven effective for DSA include decision trees (DTs) [2]–[7], neural networks [8]–[10] and support vector machines [11]–[13]. More recently, fuzzy-logic techniques [14] and ensemble learning techniques [15]–[17] have been utilized to enhance the performance of these data mining tools in security assessment of power systems. Among various data mining tools, DTs have good interpretability (or transparency) [18], in the sense that the secure operating boundary identified by DTs can be characterized by using only a few critical attributes and corresponding thresholds. As illustrated in Fig. 1, a well-trained DT can effectively and quickly produce the security classification decisions for online DSA, since only a few PMU measurements of the critical attributes are needed. The high interpretability of DTs is amenable to operator-assisted preventive and corrective actions against credible contingencies [19]. However, as discussed in [20], there exists an "accuracy versus transparency" trade-off for data mining tools. In order to obtain a more accurate classification model from DTs, one possible approach is to use an ensemble of DTs at the cost of reduced interpretability. Examples of ensembles of DTs for DSA are the multiple optimal DTs [6], random forest [15] and boosting DTs [16].

When applying data-mining-based approaches to online DSA, there are two main issues that can result in inaccurate security classification decisions. First, the realized OCs in online DSA can be dissimilar to those in the initial knowledge base prepared offline, since the predicted OCs might not be accurate and the OCs can change rapidly over time. Second, it is possible that a system topology change may occur during the operating horizon due to the forced outage of generators, transformers and transmission lines. These factors can compromise the performance of the classification model trained offline. To develop a robust data-mining-based online DSA scheme, the initial knowledge base and the classification model have to be updated in a timely manner to track these changed situations. However, there have been limited efforts directed towards handling OC variation and topology change. In the scheme proposed in [6], when the built DT fails to classify the changed OCs correctly, a new DT is built from scratch or a sub-tree of the DT is replaced by a newly built corrective DT. Aiming to deal with possible topology changes, [9] and [17]

Manuscript received October 01, 2012; revised January 24, 2013 and May 23, 2013; accepted May 25, 2013. Date of publication June 26, 2013; date of current version October 17, 2013. This work was supported in part by the US National Science Foundation under grants CPS-1035906 and CNS-1218484, in part by the DTRA grant HDTRA1-09-1-0032, and in part by the Power System Engineering Research Center. Paper no. TPWRS-01096-2012.

The authors are with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ 85287 USA (e-mail: Miao.He@asu.edu; Junshan.Zhang@asu.edu; Vijay.Vittal@asu.edu).

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

Digital Object Identifier 10.1109/TPWRS.2013.2266617

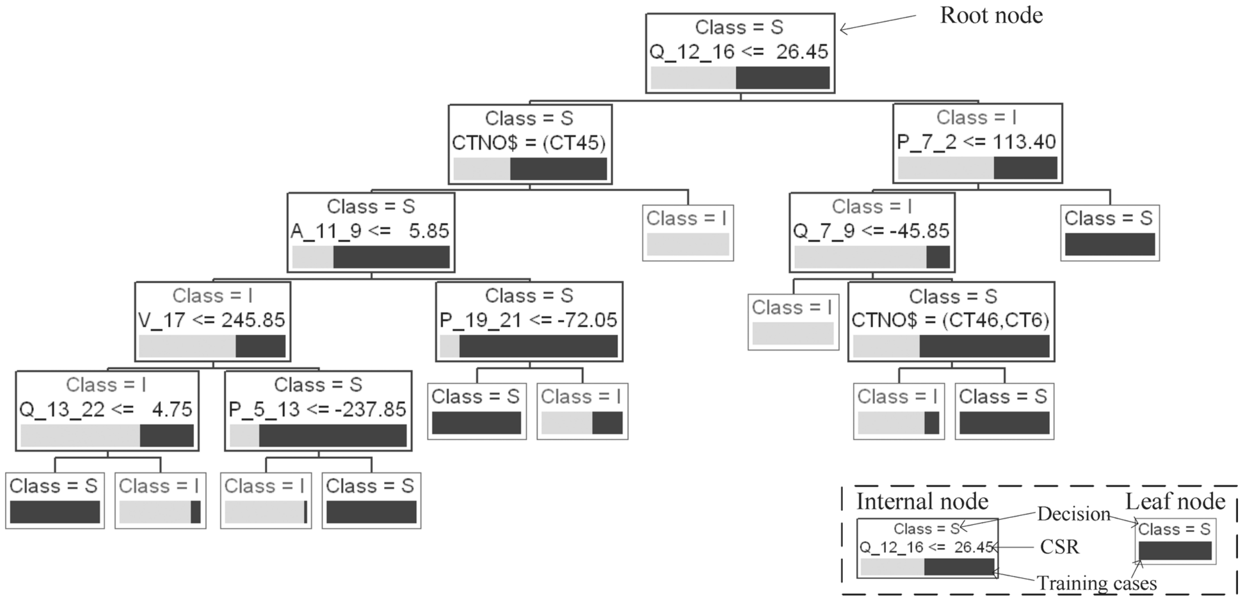


Fig. 1. Fully-grown DT of height 5 for the WECC system using an initial knowledge base consisting of 481 OCs and three critical contingencies.

suggest creating an "overall" knowledge base that covers all possible system topologies and choosing the attributes that are independent of topology for data mining.

In this paper, a robust data-mining-based DSA scheme using adaptive ensemble DT learning is proposed to handle these challenges in a more efficient manner. Specifically, the classification model for DSA is based on boosting multiple *unpruned small-height* DTs. Generally, the height of a DT is the maximal number of tests that is needed for the DT to classify a case. For the sake of brevity, small-height DTs are referred to as small DTs throughout. In offline training, the small DTs and their voting weights are sequentially identified in a "gradient-descent" manner to minimize the misclassification cost. The small DTs, together with their voting weights, are then periodically updated throughout the operating horizon by using new training cases that are created to account for any change in OC or network topology. Different from existing DT-based DSA schemes, the training cases are assigned different data weights by each small DT; and higher data weights are assigned to a new training case if it is misclassified by the small DTs. The aforementioned techniques are utilized to minimize the misclassification cost as new training cases are added to the knowledge base, so that the classification model could smoothly track the changes in OCs or system topology.

The rest of the paper is organized as follows. A brief introduction to DTs and their application to DSA are given in Section II. The proposed scheme is discussed in detail in Section III. An illustrative example by using the IEEE 39-bus test system is presented in Section IV. The proposed scheme is applied to the WECC system in Section V. Finally, conclusions are provided in Section VI.

II. Background on DT and Its Application to DSA

The data-mining framework for DSA was originally developed in [2], in which DTs were introduced to perform DSA for power systems. A DT, as illustrated in Fig. 1, is a tree-structured predictive model that maps the measurements of an attribute vector $\mathbf{x}$ to a predicted value $\hat{y}$. When DTs are used for online DSA, the attribute vector can consist of various PMU-measured variables and other system information, and the binary decision given by DTs represents the security classification decision of an OC for a critical contingency (e.g., $\hat{y} = +1$ represents the insecure case, and $\hat{y} = -1$ for the secure case). Usually, bus voltage phase angles, bus voltage magnitudes and branch power/current flows that are directly measured by PMUs are used as numerical attributes. Fig. 1 illustrates the numerical and categorical attributes used in a trained DT, in which an attribute with initial "V" stand for a bus voltage magnitude, the attributes with initials "P", "Q", and "A" stand for an active power flow, a reactive power flow, and a voltage phase angle difference between two buses, respectively (the bus numbers in attribute names are different from their real ones), "CTNO$" stands for the index of contingency.

In a DT, each non-leaf node tests the measurement of an attribute and decides which child node to drop the measurements into, and each leaf node corresponds to a predicted value. As shown in Fig. 1, in a DT for DSA, the predictive value of each leaf node is either "S" or "I", in which "S" stands for secure cases and "I" for insecure cases. Fig. 1 also illustrates the training cases that fall into each node, by using dark bars for secure cases and bright bars for insecure cases. The number of non-leaf nodes along the longest downward path from the root node to a leaf node is defined as the *height* of a DT. Given a collection of training cases $\{\mathbf{x}_n, y_n\}_{n=1}^N$, the objective of DT induction is to find a DT that can fit the training data and accurately predict the decisions for new cases. State-of-the-art DT induction algorithms are often based on greedy search. For example, in the classification and regression tree (CART) algorithm [21], the DT grows by recursively splitting the training set and choosing the critical attributes (numerical or categorical) and critical splitting rules (CSR) with the least splitting costs until some predefined stopping criterion (e.g., the size of tree or the number of training cases in a leaf node) is satisfied. In general, a fully-grown DT that accurately classifies the training cases might misclassify new cases outside the knowledge base. This feature of fully-grown DTs is usually referred to as "overfitting" [18]. In order to avoid overfitting, DTs are usually pruned by collapsing unnecessary sub-trees into leaf nodes. As illustrated in Fig. 1, in a pruned DT, some leaf nodes do not have pure training cases, which is a result of either tree pruning or early termination of tree growing [18]. By removing the nodes that may have grown based on noisy or erroneous data, the pruned DT is more resistant to overfitting than a fully-grown DT without pruning, and thus can give more accurate security decisions.

A major advancement in DT-based DSA schemes was made in [7], in which the authors proposed to build a single DT to handle multiple contingencies, by using the index of contingencies as a categorical attribute of the DT. It is worth noting that a DT built by using such an approach can give the security classification decisions of an OC concurrently for all the critical contingencies in the knowledge base, which is more efficient and can identify the critical attributes that are independent of contingencies. For example, the DT in Fig. 1, using CTNO$ as a categorical attribute, can give security classification decisions of an OC for three critical contingencies, i.e., CT6, CT45 and CT46, at the same time, and the critical attributes $Q_{12,16}$, $P_{7,2}$, $Q_{7,9}$, $A_{11,9}$, $A_{12,19}$, $A_{5,12}$ and $P_{36,7}$ can give security classification decisions independent of contingence type for some cases.

A. Small DTs

A small DT with tree height J is obtained by stopping the splitting of any leaf node if the downward path from the root node to that leaf node has exactly J non-leaf nodes. According to [22], a small DT is much less prone to overfitting compared to a fully-grown DT; therefore, the small DTs used in the proposed scheme are built without pruning. Examples of small DTs are given in Fig. 2 with $J = 2$. It can be seen that the non-leaf nodes of h_1 are exactly the same as the corresponding nodes of the DT in Fig. 1. It is worth noting that the optimal choice of J is highly dependent on the knowledge base, and should be decided based on a bias-variance analysis [18], which will be discussed in the case study of Section IV. Note also that different from [18], the tree height, instead of the number of nodes, is used as the metric to quantify the tree size. The reason, which will be soon apparent, is to restrict the number of nodes that will be revised when updating DTs to a value less than J.

B. Ensemble of DTs

In ensemble-DT-based DSA schemes, the security classification decision of an OC vector $\mathbf{x}$, denoted by $H_L(\mathbf{x})$, is made based on the voting of multiple DTs. For an ensemble of DTs $h_l(l = 1, 2, \ldots, L)$, there are two approaches to DSA classification: deterministic and probabilistic. For the deterministic approach, the security classification decision is given by

$$H_L(\mathbf{x}) = \begin{cases} +1, & \text{if } \sum_{l=1}^{L} a_l h_l(\mathbf{x}) \geq 0 \\ -1, & \text{o.w.} \end{cases} \tag{1}$$

where a_l $(l = 1, 2, \ldots, L)$ are the voting weights of DTs. To obtain probabilistic classification decisions, the "logistic correction" technique [23] can be applied. Then, the probability of an "Insecure" classification decision is given by

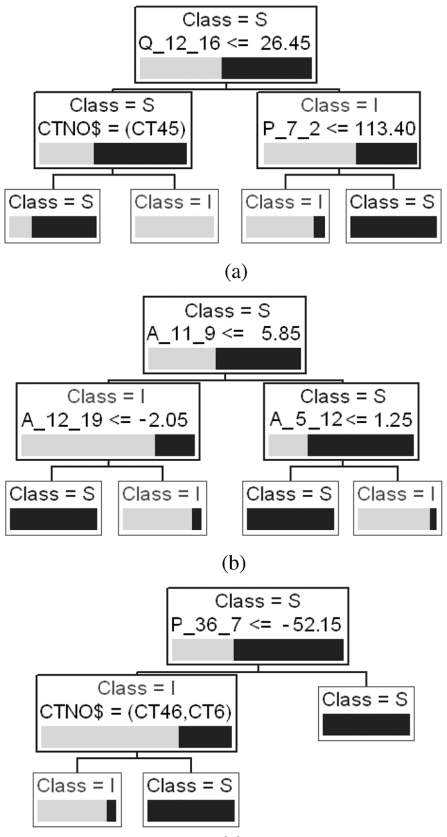


Fig. 2. First three small DTs ($J = 2$) for the WECC system, the voting weights of which are 4.38, 3.04 and 0.93, respectively. (a) Small DT h_1. (b) Small DT h_2. (c) Small DT h_3.

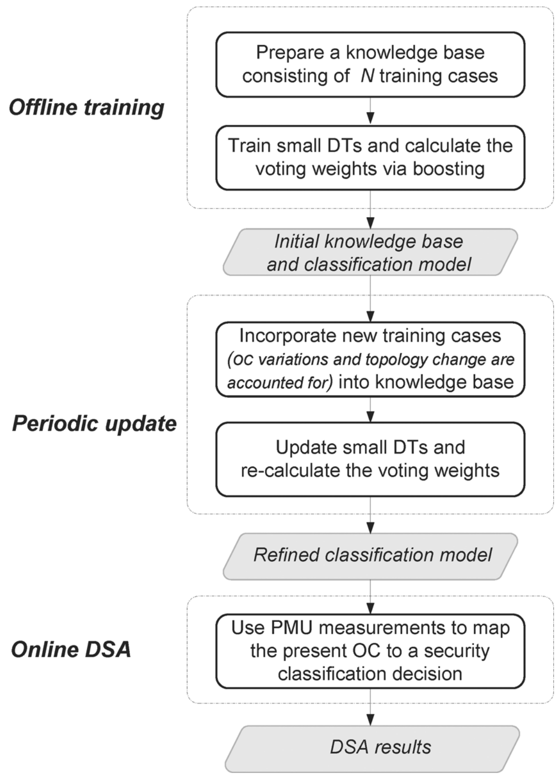
$$\Pr(H_L(\mathbf{x}) = +1 \mid \mathbf{x}) = \frac{1}{1 + \exp\left(-\sum_{l=1}^{L} a_l h_l(\mathbf{x})\right)}. \tag{2}$$

In this paper, deterministic classification decision is used to calculate the misclassification rate for case studies.

The existing methods for ensemble DT learning include bagging, random subspace method, boosting and random forest. Reference [24] compares these methods, and finds that boosting and random forest achieve significantly better performance than the others. In previous work by the authors [16], an algorithm for boosting DTs is developed in the context of avoiding overfitting to noisy training data. In this paper, the boosting algorithm is employed in online DSA to deal with OC variations and possible topology changes. The algorithm for building the small DTs and calculating the voting weights will be discussed in Section III-A.

C. Updating DTs

One existing approach for updating a DT without rebuilding it from scratch is the efficient tree restructuring algorithm [25], with the main idea summarized as follows. When incorporating a new case, the DT remains unchanged if the new case is classified correctly; otherwise, the non-leaf nodes along the path



Offline training
Prepare a knowledge base consisting of N training cases
Train small DTs and calculate the voting weights via boosting
Initial knowledge base and classification model
Periodic update
Incorporate new training cases (OC variations and topology change are accounted for) into knowledge base
Update small DTs and re-calculate the voting weights
Refined classification model
Online DSA
Use PMU measurements to map the present OC to a security classification decision
DSA results

Fig. 3. Proposed scheme for online DSA using adaptive ensemble DT learning.

which the new case passes are revised in a top-down manner. Specifically, for each non-leaf node to be revised, a new test is first identified by using the new case as well as the existing cases that fall into the non-leaf node. If different from the original test, the newly identified test is then installed at the non-leaf node, followed by tree restructuring operations recursively applied on the sub-tree corresponding to that non-leaf node (there are six slightly different restructuring operations for various structures of the sub-tree, which are not discussed here). The motivation for these restructuring operations is that the original test at the non-leaf node is highly likely to be the optimal tests for the two child nodes after restructuring, which is usually the case when categorical attributes are used by the test [25]; in this scenario, the two child nodes are exempted from further update.

III. Proposed Online DSA Scheme

The proposed scheme for online DSA, as illustrated in Fig. 3, consists of three steps, with the details described below.

A. Offline Training

1) Initial Knowledge Base Preparation: First, $N_{\rm OC}$ predicted OCs are generated day ahead for each period of the future operating horizon (e.g., the next 24 hours) based on day-ahead load forecast and generation schedules; each period may span several hours, and can be divided according to the hours of peak load, shoulder load and off-peak load. Then, for each of the $N_{\rm OC}$ day-ahead predicted OCs, detailed power flow analysis and time-domain simulations are performed for K critical contingencies that are selected by the system operator or based on prior experience. It is worth noting that the key focus here is on dealing with OC variations and possible topology change, and thus the selection or screening of critical contingencies is beyond the scope of this study. By using specified dynamic security criteria (e.g., transient stability, damping performance, transient voltage drop/rise, transient frequency, relay margin), the day-ahead predicted OCs are labeled as "Secure" or "Insecure" for each critical contingency.

As a result, an initial knowledge base that consists of $N = N_{\rm OC} \times K$ training cases is obtained, in which each case is represented by a vector $\{x_1, \ldots, x_P, y\}$, where x_1 is the index of a critical contingency, $\{x_2, \ldots, x_P\}$ are the values of numerical attributes obtained from power flow analysis of an OC, and y is the transient security classification decision of the OC for the critical contingency x_1. Based on the previous studies [5]–[7], the following PMU-measured variables are selected as numerical attributes:

- Branch active power flows $\{P_{ij}; i \in \mathcal{B} \text{ or } j \in \mathcal{B}\}$
- Branch reactive power flows $\{Q_{ij}; i \in \mathcal{B} \text{ or} j \in \mathcal{B}\}$
- Branch current flows (magnitude) $\{I_{ij}; i \in \mathcal{B} \text{ or } j \in \mathcal{B}\}$
- Bus voltage magnitudes $\{V_i; i \in \mathcal{B}\}$
- Bus voltage phase angle differences $\{A_{ij} \triangleq A_i - A_j; i, j \in \mathcal{B} \text{ and } i > j\}$

where $\mathcal{B}$ denotes the set of PMU buses in the system. It is worth noting that only raw measurements reported by PMUs are used as the numerical attributes in this work; more generally, the variables computed using other system information may also be used, e.g., the voltage at the bus connected to a PMU bus when the branch impedance is constant [5].

2) Boosting Small DTs: The basic algorithmic flowchart of boosting small DTs is illustrated in Fig. 4. For convenience, define $\mathcal{H}_J$ as the class of small DTs with height J, define F_L as the score of the weighted voting of the ensemble of small DTs, i.e., $F_L(\mathbf{x}) = \sum_{l=1}^{L} a_l h_l(\mathbf{x})$, and define $C_N(F_L)$ as the cost function of F_L on the N training cases, given by

$$C_N(F_L) = \frac{1}{N}\sum_{n=1}^{N} \log_2(1 + \mathrm{e}^{-y_n F_L(\mathbf{x_n})}). \tag{3}$$

It is observed from (1) and (3) that $C_N(F_L)$ lies strictly above the misclassification error rate of H_L. Then, a primary objective of boosting is to minimize $C_N(F_L)$, by identifying the small DTs $h_l \in \mathcal{H}_J$ and their voting weights $a_l \in \mathcal{R}^+$. An analytical formulation is provided as follows:

$$\mathcal{P}_F : \min_{\substack{h_1,\ldots,h_L \in \mathcal{H}_J \\ a_1,\ldots,a_L \in \mathcal{R}^+}} C_N(F_L). \tag{4}$$

The convexity and the differentiability of $C_N(F_L)$ with regard to F_L make it possible to solve $\mathcal{P}_F$ in (4) by using a line search strategy [26], the details of which are summarized as follows. A small DT h_l is chosen to be the "gradient" of C_N at F_{l-1} projected onto $\mathcal{H}_J$, and the voting weight a_l is computed as the "step size" that minimizes $C_N(F_{l-1}+a_l h_l)$. Then, the small DT h_l is added to F_{l-1} to obtain $F_l = F_{l-1} + a_l h_l$. The above steps are iterated, for $l = 1, 2, \ldots, L$, by using F_0 as a zero function. More specifically, it is shown in [16] that the small DT h_l can be obtained by solving the following problem:

$$\mathcal{P}_h^{(l)} : \min_{h_l \in \mathcal{H}_J} \frac{1}{N}\sum_{n=1}^{N} w_n^{(l)} \mathbf{1}_{\{y_n \neq h_l(\mathbf{x}_n)\}} \tag{5}$$

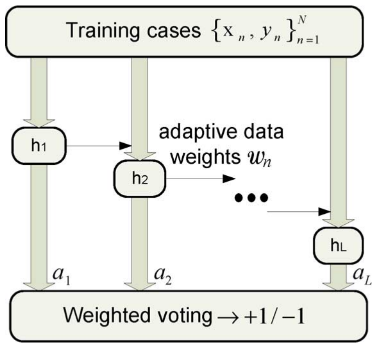

Training cases $\{x_n, y_n\}_{n=1}^N$
h1
h2
hL
adaptive data weights w_n
a_1
a_2
a_L
Weighted voting → +1/−1

Fig. 4. Boosting small DTs.

where $w_n^{(l)} \triangleq (1+e^{y_n F_{l-1}(\mathbf{x}_n)})^{-1}$ is the positive data weight of the training case $\{\mathbf{x}_n, y_n\}$, and $\mathbf{1}_{\{y_n \neq h_l(\mathbf{x}_n)\}}$ takes value 0 if the training case $\{\mathbf{x}_n, y_n\}$ is correctly classified by the small DT h_l (otherwise, it takes value 1). By definition of $w_n^{(l)}$, it is easy to observe that the data weights are assigned *adaptively* by small DTs, in the sense that if the training case $\{\mathbf{x}_n, y_n\}$ is misclassified by the small DT h_l, then $w_n^{(l+1)} > w_n^{(l)}$, i.e., the training case has a higher data weight in the next round of the boosting process. It is worth noting that highly skewed training data (e.g., the case in [15]) can be handled by scaling up the weights of under-represented cases, such that $\sum_{y=+1} w_n^{(l)} = \sum_{y=-1} w_n^{(l)}$. As suggested in (5), the objective of $\mathcal{P}_h^{(l)}$ is to determine the small DT that has the least misclassification error rate on the weighted training data. Thus, the small DT h_l can be obtained by employing the standard CART algorithm [21] subject to the tree height J, and by using misclassification error rate as the splitting cost when building the DT. Then, its positive voting weight is obtained by solving the following problem:

$$\mathcal{P}_a^{(l)} : \min_{a \in \mathcal{R}^+} G_N^{(l)}(a) \tag{6}$$

where $G_N^{(l)}(a) \triangleq C_N(F_{l-1} + a h_l)$. Under the condition that h_l is a "descent direction" of $C_N(F_{l-1})$, it is easy to verify that $G_N^{(l)'}(0) < 0$ and $G_N^{(l)''}(a) > 0$ holds for any $a \in \mathcal{R}^+$. Therefore, $G_N^{(l)}(a)$ has a unique minimum in $\mathcal{R}^+$ that can be found using standard numerical solution methods (e.g., Newton's method).

B. Periodic Updates

1) New Training Case Creation: In the initial knowledge base prepared offline, the predicted OCs generated using day-ahead forecast may not reflect the actual system conditions, which is very likely to be the case for power systems with high penetration of variable renewable generation and distributed generation. Therefore, as the operating horizon is approached and the data available to system operators is updated, it will be necessary to utilize short-term forecast and schedules to generate newly changed OCs and add them to the knowledge base on a slot-by-slot basis (one slot may span several minutes depending on the processing speed [5]). Further, in case of topology change, the post-disturbance OCs should also be incorporated into the knowledge base. After power flow analysis of these newly changed OCs, new training cases are generated as described in Section III-A1. It is worth noting that during the operating horizon, it is also likely that the knowledge base may need to be updated by incorporating new contingencies of interest. The solution to this problem has been discussed in [16]. In this work, the critical contingency list is assumed to remain unchanged during the operating horizon.

2) Updating the Classification Model: Given the newly created training cases, the classification model is updated by using one new case at a time. Specifically, for the kth new training case $\{\mathbf{x}_{N+k}, y_{N+k}\}$, the classification model is updated by incorporating $\{\mathbf{x}_{N+k}, y_{N+k}\}$ with a data weight $w_{N+k}^{(l)} = (1 + e^{y_{N+k} F_{l-1}(\mathbf{x}_{N+k})})^{-1}$ into the small DT h_l and recalculating the voting weight a_l, iteratively for $l = 1, 2, \ldots, L$.

A key step for incorporating a new training case into a small DT is to adopt the method described in Section II-C. Since misclassification error rate is used as the metric of splitting cost, as suggested in (5), it is easy to observe that there exists a even simpler solution for updating the small DTs. Specifically, a small DT remains unchanged if the new case is correctly classified; therwise, only the sub-tree corresponding to the first non-leaf node that has a different decision for the new case is subject to update. It is worth noting that, since the tree height is J, the total number of non-leaf nodes to be revised is at most J. After the small DT h_l is updated, its voting weight a_l is recalculated by minimizing $G_{N+k}^{(l)}(a)$.

The process of updating the classification model is summarized in Algorithm 1. It is useful to note that when the kth new training case is used to update the small DTs, the data weights of the previous $N+k-1$ training cases calculated in Step 4 of Algorithm 1 are different from the data weights that were used in building or updating the small DTs in the past rounds. Therefore, unlike the case in offline training, it is possible that the updated small DT h_l is not a "descent direction" of C_{N+k} at F_{l-1} any more. In order to detect and handle this situation, an extra step is used in Algorithm 1. Specifically, if $\sum_{n=1}^{N+k} w_n^{(l)} y_n h_l(\mathbf{x}_n) < 0$, then $-h_l$ is a "descent direction" and used for weighted voting.

Algorithm 1: Periodic updates using a new training case

```
1:  Input: A new training case $\{\mathbf{x}_{N+k}, y_{N+k}\}$.
2:  Initialization: $F_0 = \mathbf{0}$.
3:  for $l = 1$ to $L$ do
4:      Recalculate the data weights of $\{\mathbf{x}_n, y_n\}_{n=1}^{N+k-1}$.
5:      Incorporate $\{\mathbf{x}_{N+k}, y_{N+k}\}$ with weight $w_{N+k}^{(l)}$ into
        $h_l$.
6:      Calculate $\varepsilon = \frac{1}{N+k} \sum_{n=1}^{N+k} w_n^{(l)} y_n h_l(\mathbf{x}_n)$.
7:      if $\varepsilon < 0$ then
8:          $h_l \leftarrow -h_l$.
9:      end if
10:     Recalculate $a_l$ by minimizing $G_{N+k}^{(l)}(a)$.
11:     $F_l \leftarrow F_{l-1} + a_l h_l$.
12: end for
```

C. Online DSA

In real-time, when the synchronized PMU measurements are received, the pre-fault values of the numerical attributes are retrieved and combined with the indices of all critical contingencies to create K unlabeled cases, which will be used by the classification model to give security classification decisions of the present OC for the K critical contingencies. Specifically, when an unlabeled case is processed by the classification model, each of the small DTs uses the values of the attribute vector and its CSRs to produce a binary decision. Finally, the binary decisions of all small DTs are collected and used to give the security classification decisions of the present OC, according to (1). It is worth noting that distributed processing technologies [27] can be leveraged to speed up online DSA. Specifically, the K unlabeled cases can be classified separately by using K duplicates of the classification model, and in each classification model, all small DTs can process the attribute vector of an unlabeled case in a parallel manner.

From the above development, it can be seen that the proposed scheme illustrated in Fig. 3 is derived from those in previous work [5]–[7], with the following major modifications. 1) The classification model is obtained via boosting multiple small unpruned DTs instead of a single fully-grown DT after pruning. It is suggested that boosting algorithms can lead to better model fitting and the produced classification model is quite resistant to overfitting [22]. Thus, boosting small DTs has great potential to deliver better performance in terms of classification accuracy. 2) Unequal data weights are assigned to the training cases adaptively by small DTs. In periodic updates, misclassified new training cases can have higher data weights than those classified correctly. This will speed up adapting the small DTs to newly changed OCs. 3) The small DTs are gracefully updated by incorporating new cases one at a time, whereas rebuilding DTs is used in [5]–[7]. 4) The DT and the knowledge base are updated only when the new cases are misclassified in [5]–[7]; whereas all new training cases are incorporated into the knowledge base in the proposed scheme.

IV. Illustrative Example

The IEEE 39-bus test system [28] is used as an illustrative small system. As illustrated in Fig. 5, 8 PMUs are installed in the system, according to the placement design provided in [29]. In what follows, the main steps of the proposed approach, including attribute selection, knowledge base preparation and ensemble small DT learning, will be demonstrated by using the IEEE 39-bus test system. Finally, the results of robustness test on changed OCs will be presented.

A. Knowledge Base

1) Attribute Selection: Based on the PMU placement and system topology in Fig. 5, 111 numerical attributes are selected according to the rules described in Section III-A, including:

- 8 bus voltage magnitudes at the 8 PMU buses;
- 75 branch active/reactive power flows and current flows, which take any of the 8 PMU buses as either a from-bus or a to-bus of the branch;
- 28 bus voltage phase angle differences, which are computed from the $(8(8-1))/(2)$ pairs of phase angles.

2) OC Generation and Contingencies: The OC specified in [28] is used as the base OC. To enrich the knowledge base, more OCs are generated by randomly changing the bus loads (both active and reactive) within 90% to 110% of their original values

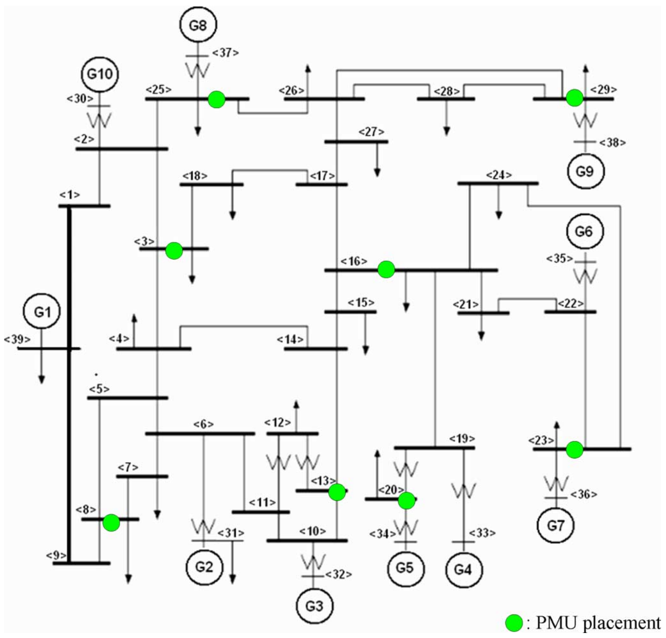


Fig. 5. IEEE 39-bus system with 8 PMUs.

in the base OC. For each generated OC, limit checking is carried out by using the power flow and short circuit analysis tool (PSAT) [30], so that any generated OC with pre-contingency overloading or violation of voltage magnitude/angle limits is not included in the knowledge base. Further, transient stability assessment is carried out for the 30 $N-2$ contingencies listed in [31, Table II]. These $N-2$ contingencies, which can lead to stressed system conditions, are identified by exhaustive search among all possible $N-2$ contingencies.

3) Transient Stability Assessment Tool and Criteria: The transient security assessment tool (TSAT) [30] is used to assess the transient performance of the generated OCs. The time-domain simulation is executed for 10 s with a step size of 0.5 cycle. The power angle-based stability margin is used as the transient stability index (TSI), defined as

$$\eta = \frac{360-\delta_{\max}}{360+\delta_{\max}} \times 100, \quad -100 < \eta < 100 \tag{7}$$

where $\delta_{\max}$ is the maximum angle separation of any two generators in the system at the same time in the post-fault response. In case of islanding, the above value is evaluated for each island and the smallest value is taken as the TSI. During the simulation time, whenever the margin η turns out to be negative, i.e., the rotor angle difference of any two generators exceeds 360 degree, the case is labeled as transiently insecure.

B. Offline Training

1) Choice of J and L: V-fold cross validation $(V=10)$ is carried out to determine the optimal tree height J and the optimal number of small DTs L. Specifically, the training cases in the initial knowledge base are randomly partitioned into V subsets of equal size. For given fixed J and L, a classification model is trained by using $V-1$ subsets, and tested using the other subset. The training process is then repeated V times in total, with each of the V subsets used exactly once as the test

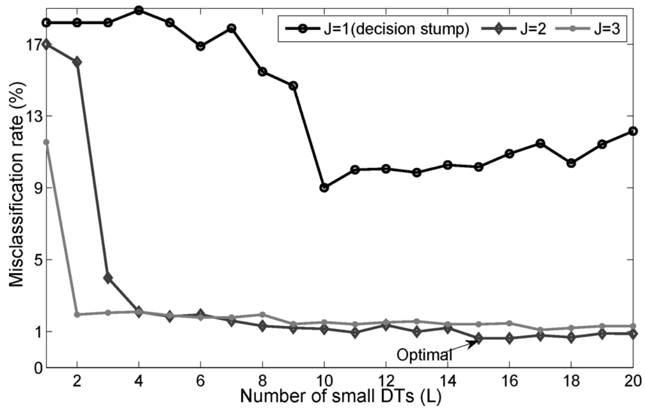


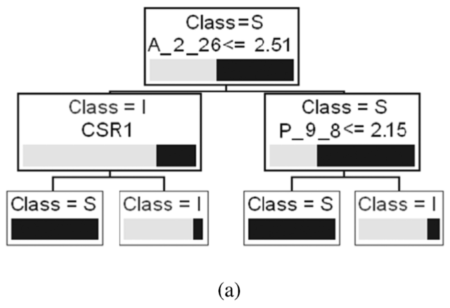
Fig. 6. Ensemble small DT learning with different tree heights for the IEEE 39-bus test system.

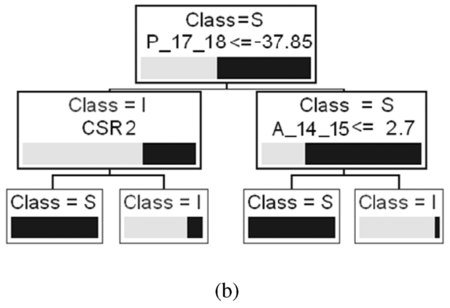
data. Finally, the misclassification error rate obtained by V-fold cross validation is calculated by averaging over the V classification models. The results of the above procedure for different tree heights $(J = 1, 2, 3)$ are illustrated in Fig. 6. It can be seen that as L increases, the misclassification error rate of each classification model decreases and reaches a plateau at some L. Then, when L grows larger, each classification model incurs a larger variance and hence a higher misclassification error rate. On the other hand, a larger tree height J implies a larger variance of classification model [18], which is also observed in Fig. 6. Based on these observations, $J = 2$ is chosen, and $L = 15$ at which the misclassification error rate drops below 1% and reaches a plateau is selected.

2) Ensemble Small DT Learning: When the optimal tree height J and the optimal number of small DTs L are determined, the algorithm described in Section III-A2 is used to build the ensemble of small DTs. Specifically, for $l = 1, 2, \ldots, L$, the data weights $w_n^{(l)}$ are first computed according to (5). Then, the training cases together with their data weights are used by the CART algorithm to build a small DT h_l with height J, by using weighted misclassification rate as the cost function, as shown in (5). Note that *each small DT gives security classification decisions for all critical contingencies*. Further, the voting weight of h_l is calculated by numerically solving (6). Then, the ensemble of small DTs are obtained. It is worth noting that, different from the V-fold cross validation procedure, the entire training set (not a subset) is used by each small DT of the ensemble.

C. Robustness Testing

1) Changed OCs: In the IEEE 39-bus test system, generator **G1**, together with transmission lines (39, 9) and (39, 1), represents the equivalent to the external system of the New England area [28]. It is now assumed that the capacity of **G1** reduces from 1100 MW to 900 MW, which could be the result of either the loss of a transmission corridor or a generator tripping outside the New England area. Therefore, the OCs will change due to generation rescheduling. By setting the capacity of **G1** to 900 MW, changed OCs are generated by rescheduling generation and re-solving power flows for each OC in the initial knowledge base. These changed OCs will be utilized to test the robustness of the proposed approach.





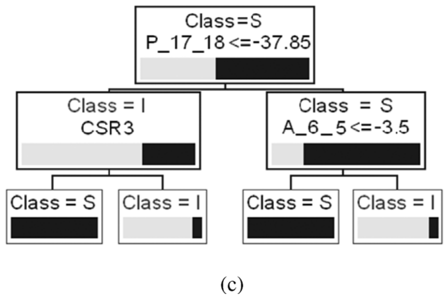


Fig. 7. First small DT h_1 $(J = 2)$ for the IEEE 39-bus test system. (a) Trained small DT h_1 [CSR1 represents the critical splitting rule: CTNO\$ = (CT1, CT2, CT3, CT7, CT11, CT12, CT15, CT16, CT17)]. (b) Small DT h_1 updated with changed OCs [CSR2 represents: CTNO\$ = (CT1, CT2, CT3, CT4, CT7, CT8, CT12, CT15, CT16, CT19)]. (c) Small DT h_1 rebuilt with changed OCs [CSR3 represents: CTNO\$ = (CT1, CT2, CT3, CT4, CT7, CT9, CT11, CT15, CT17)].

2) Robustness Testing Results: First, 200 OCs are generated to create the initial knowledge base consisting of 6000 (200 OCs $\times$ 30 contingencies) training cases. Accordingly, another 200 changed OCs are generated, in which 100 OCs are used to update the small DTs and the other 100 OCs are used for robustness testing. In the proposed approach, Algorithm 1 is applied to update each of the 15 small DTs by using the 3000 (100 OCs $\times$ 30 contingencies) new cases. To illustrate the change of small DTs, the first small DT h_1 is used as an example. Specifically, h_1 obtained in offline training and updated with the 100 changed OCs by using the proposed approach are illustrated in Fig. 7(a) and (b), respectively. It is observed that due to the changed OCs and generation rescheduling, the critical attribute in the root node of h_1 changes from the voltage phase angle difference between bus 2 and bus 26, A_2_26, to the active power flow between bus 17 and bus 18, P_17_18. The CSRs of the non-root nodes change accordingly, as a result of the recursive procedure of the CART algorithm. The small DT h_1 rebuilt with the 100 changed OCs is illustrated in Fig. 7(c), which has the same CSR at the root node as the small DT updated by using

TABLE I
MISCLASSIFICATION ERROR RATE OF ROBUSTNESS TESTING

	secure cases	insecure cases	overall
Proposed	0.68%	0.36 %	0.55%
Small DTs (rebuilding)	0.59%	0.38%	0.54%
Small DTs (no updating)	10.68%	6.85%	9.57%

the proposed approach. Since the small DTs h_1 obtained by updating and rebuilding are different at non-root nodes, the other small DTs, h_2 to h_{15} are also different. This is because the ensemble DT learning algorithm sequentially updates/builds the small DTs, in which each small DT depends on the previous small DTs.

The proposed approach is compared with two benchmark approaches: 1) small DTs rebuilt by using the 100 changed OCs together with the initial 200 OCs, 2) small DTs without updating. The test results of the three approaches are presented in Table I. It can be seen that the proposed approach achieves comparable performance to the benchmark approach by rebuilding small DTs. The test results also suggest that when OCs change, the small DTs have to be updated in order to track the variation of OCs.

V. APPLICATION TO THE WECC SYSTEM

The test power system used in this case study is part of the Western Electricity Coordinating Council (WECC) system. It consists of over 600 buses (of which 33 are PMU buses), 700 transmission lines and 100 generators.

A. Knowledge Base

1) OC Generation: The OCs used in the case study are generated by using real-life data of power flows, bus loads and generator power outputs that were recorded every 15 min during a 2008 summer peak day. The overall load profile is illustrated in Fig. 8. Based on the variations of the aggregate load, each period for offline training is chosen to span 8 hours, and the peak load period 12:00 Hrs–20:00 Hrs is investigated in this case study. Basically, there are three sets of generated OCs used in this case study: day-ahead predicted OCs, short-term predicted OCs and realized OCs. The day-ahead predicted OCs are used to create the initial knowledge base, the short-term predicted OCs are used to create the new training cases to update the knowledge base and the classification model, and the realized OCs are used for testing purposes only.

In what follows, the procedure for generating the three OC sets is discussed in detail. The realized OCs include the 33 recorded OCs and another 448 OCs that are generated by interpolation, as illustrated in Fig. 8. Specifically, following the method in [7], both the active and reactive load of each load bus for every minute of the investigated period are obtained by linear interpolation based on the two closest recorded OCs, and the generator power outputs are adjusted as needed to ensure valid OCs. To enrich the initial knowledge base, a day-ahead predicted OC is obtained by randomly changing the bus loads within 90% to 110% of the loads of the corresponding realized OC, by using a uniform distribution. Similarly, a short-term predicted OC is generated by uniformly randomly changing the bus

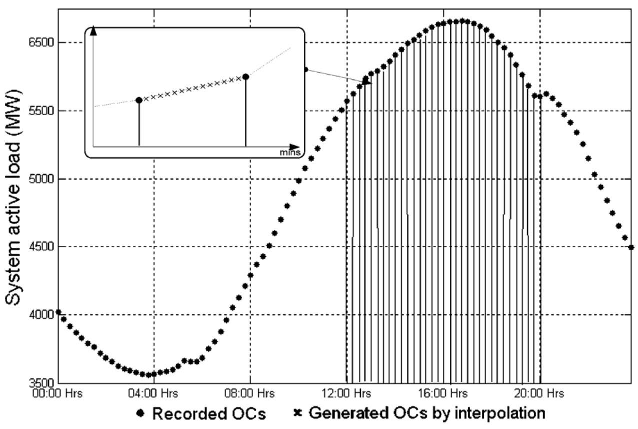


Fig. 8. Aggregate load of recorded OCs and generated OCs by interpolation.

loads within 97% to 103% of the loads of the corresponding realized OC. After solving the power flows for each OC using the power flow and short circuit analysis tool (PSAT) [30], 481 OCs are generated for each of the three OC sets. It is worth noting that different from the day-ahead predicted OCs, the short-term predicted OCs and the realized OCs are time-stamped.

2) Critical Contingency Selection: A contingency list, which was created by the regional grid operator to account for possible outages of transmission lines, three-winding transformers and generators that could have significant impact, is used here. Specifically, the contingency list consists of 1 $N-4$ contingency, 8 $N-3$ contingencies, 172 $N-2$ contingencies, and 0 $N-1$ contingencies (i.e., no $N-1$ contingencies lead to insecure conditions). The power angle-based stability margin defined in (7) is used as the transient stability index. After performing transient security assessment by using TSAT for all realized OCs and adhering to the above security criteria, three $N-2$ contingencies which lead to transiently insecure cases are selected as the critical contingencies in the knowledge base. Each of the three $N-2$ critical contingencies is initiated by a "three-phase short circuit to ground" fault at a bus which is cleared after 5 cycles, by tripping a transmission line that connects the bus and by disconnecting a generator that will go out of step as a result of the line tripping.

3) Case Creation: Combining the three sets of generated OCs with their transient security classification decisions for the three critical contingencies, $N = 1443$ cases are created for the initial knowledge base, for updating and for testing, respectively. Based on the interconnection structure of the 33 PMU buses, 799 numerical attributes are identified using the rules described in Section III.A; thus $P = 800$. For each case, the values of the 799 numerical attributes are obtained from the power flow solutions. Then, the initial knowledge base is organized into an $N \times (P+1)$ array.

B. Offline Training

The initial knowledge base as an $N \times (P+1)$ array is first used by the CART algorithm to build the small DTs. Following the procedure described in Section III-D, it is found that $J = 2$ and $L = 35$ give the best results of V-fold cross validation. The first three small DTs built from the initial knowledge base are illustrated in Fig. 2. For comparison, a fully-grown single

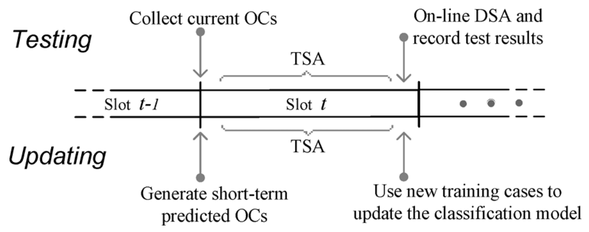

Testing
Collect current OCs
On-line DSA and record test results
TSA
Slot t-1
Slot t
TSA
Updating
Generate short-term predicted OCs
Use new training cases to update the classification model

Fig. 9. Flowchart for testing online DSA with periodic updates.

DT with pruning is also built, as illustrated in Fig. 1 (in order to give a concrete impression of DTs and small DTs used for DSA, Figs. 1 and 2 were presented in Section II).

C. Online DSA Simulation

The online DSA is simulated iteratively on a slot-by-slot basis, as illustrated in Fig. 9. Generally, each slot spans M minutes. Since it is sufficient to perform security assessment of a short-term predicted OC for the three $N-2$ critical contingencies, $M=1$ is chosen here. In case of more critical contingencies or a larger test system, a longer slot can be chosen. In online DSA, a third scheme in which the classification model is obtained by boosting small DTs but updated by rebuilding is compared with the two aforementioned schemes.

1) OC Variations in Sub-Period 12:00 Hrs–16:00 Hrs: In each slot of this sub-period, the $3M$ test cases created from the M realized OCs with time-stamps falling into this slot are collected, and then used as the present OCs for online DSA to assess the performance of the classification model updated so far. Meanwhile, another $3M$ new training cases created from the short-term predicted OC for the next slot are incorporated into the knowledge base to update the classification model.

2) Topology Change in Sub-Period 16:00 Hrs–20:00 Hrs: At the peak hour 16:00 Hrs, a topology change is imposed on the test system, and assumed to last for the remaining hours of the day. Specifically, among the 178 contingencies that do not incur transient instability for all realized OCs, the contingency which has the least positive margin averaged over all realized OCs is chosen; as a result, a transmission line is removed and a generator is disconnected from the test system. Then, the new training cases and test cases during the latter sub-period are created using an approach similar to those used in the former sub-period, but by using a different system topology.

D. Test Results and Discussion

Throughout the entire horizon of the above online DSA simulations, the misclassification error rate and the computation time for updating in each slot are recorded and summarized in Table II and Fig. 10, respectively.

1) Classification Accuracy: As illustrated in Table II, the two boosting-based schemes turn out to be more accurate than the single-DT-based scheme for both simulation sub-periods, and the performance of the proposed scheme is quite close to the scheme based on boosting small DTs with rebuilding.

2) Computation Requirement: The computation time required by updating the classification models using new OCs is illustrated in Fig. 10. It is clear that the proposed scheme requires the lowest computation time. Further, as the number of new OCs increases, the proposed scheme becomes less time-consuming than the other two schemes. The reason is that for each new OC, the two benchmark schemes rebuild DTs from scratch, while the graceful update of small DTs is carried out in the proposed scheme. Further, according to the CART algorithm [21], it is known that the sorting operation of the CART algorithm dominates the computational burden of DT building/rebuilding. When updating small DTs, the sorting operation is skipped [25]. Therefore, the proposed scheme has a much lower computational burden.

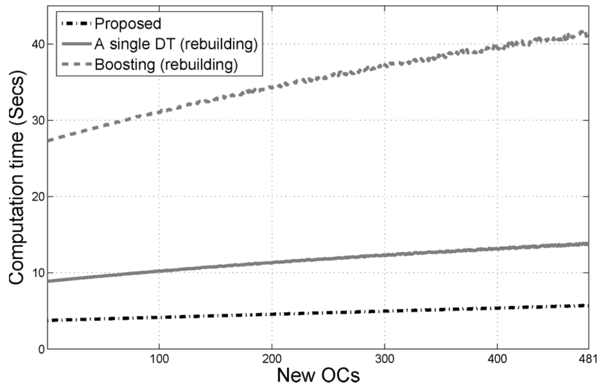


Fig. 10. Computation time for updating/rebuilding (executed in MATLAB on a workstation with an Intel Pentium IV 3.20-GHz CPU and 4 GB of RAM).

TABLE II
MISCLASSIFICATION ERROR RATE OF ONLINE DSA

Scheme	Sub-period 12-16 Hrs			Sub-period 16-20 Hrs		
	secure cases	insecure cases	overall	secure cases	insecure cases	overall
Proposed	2.41%	1.03%	1.67%	2.54%	1.08%	1.74%
A single DT (rebuilding)	2.71%	1.80%	2.22%	2.26%	2.73%	2.5%
Boosting (rebuilding)	1.81%	1.03%	1.39%	2.26%	0.82%	1.5%

VI. CONCLUSION

In this study, a data-mining-based online DSA scheme is proposed to handle the OC variations and topology change that are likely to occur during the operating horizon. The proposed scheme is applied to a practical power system, and the results of a case study demonstrate the performance improvement brought by boosting unpruned small DTs over a single DT. Compared to single DTs, the classification model obtained from ensemble DT learning often have higher accuracy, and lend themselves to cost-effective incorporation of new training cases. The results presented here also provide an insight into the possibilities of other ensemble DT learning techniques, e.g., random forest, in handling the challenges of online DSA.

ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. The first author is grateful to Dr. R. Diao at Pacific Northwest National Laboratory for his help with the case study.

References

[1] P. Sauer, K. L. Tomsovic, and V. Vittal, *Dynamic Security Assessment*, ser. The Electric Power Engineering Handbook, 2nd ed. Boca Raton, FL, USA: CRC, 2007, ch. 15, pp. 1–10.

[2] L. Wehenkel, T. Van Cutsem, and M. Ribbens-Pavella, "An artificial intelligence framework for online transient stability assessment of power systems," *IEEE Trans. Power Syst.*, vol. 4, no. 2, pp. 789–800, May 1989.

[3] L. Wehenkel, M. Pavella, E. Euxibie, and B. Heilbronn, "Decision tree based transient stability method: A case study," *IEEE Trans. Power Syst.*, vol. 9, no. 1, pp. 459–469, Feb. 1994.

[4] S. Rovnyak, S. Kretsinger, J. Thorp, and D. Brown, "Decision trees for real-time transient stability prediction," *IEEE Trans. Power Syst.*, vol. 9, no. 3, pp. 1417–1426, Aug. 1994.

[5] K. Sun, S. Likhate, V. Vittal, V. 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.

[6] R. Diao, K. Sun, V. Vittal, R. O'Keefe, M. Richardson, N. Bhatt, D. Stradford, and S. Sarawgi, "Decision tree-based online voltage security assessment using PMU measurements," *IEEE Trans. Power Syst.*, vol. 24, no. 2, pp. 832–839, May 2009.

[7] R. Diao, V. Vittal, and N. Logic, "Design of a real-time security assessment tool for situational awareness enhancement in modern power systems," *IEEE Trans. Power Syst.*, vol. 25, no. 2, pp. 957–965, May 2010.

[8] V. Miranda, J. Fidalgo, J. Lopes, and L. Almeida, "Real time preventive actions for transient stability enhancement with a hybrid neural network-optimization approach," *IEEE Trans. Power Syst.*, vol. 10, no. 2, pp. 1029–1035, May 1995.

[9] C. Jensen, M. El-Sharkawi, and R. Marks, "Power system security assessment using neural networks: Feature selection using Fisher discrimination," *IEEE Trans. Power Syst.*, vol. 16, no. 4, pp. 757–763, Nov. 2001.

[10] I. Kamwa, R. Grondin, and L. Loud, "Time-varying contingency screening for dynamic security assessment using intelligent-systems techniques," *IEEE Trans. Power Syst.*, vol. 16, no. 3, pp. 526–536, Aug. 2001.

[11] L. Moulin, A. da Silva, M. El-Sharkawi, I. Marks, and R. J. , "Support vector machines for transient stability analysis of large-scale power systems," *IEEE Trans. Power Syst.*, vol. 19, no. 2, pp. 818–825, May 2004.

[12] A. Rajapakse, F. Gomez, K. Nanayakkara, P. Crossley, and V. Terzija, "Rotor angle instability prediction using post-disturbance voltage trajectories," *IEEE Trans. Power Syst.*, vol. 25, no. 2, pp. 947–956, May 2010.

[13] F. Gomez, A. Rajapakse, U. Annakkage, and I. Fernando, "Support vector machine-based algorithm for post-fault transient stability status prediction using synchronized measurements," *IEEE Trans. Power Syst.*, vol. 26, no. 3, pp. 1474–1483, Aug. 2011.

[14] I. Kamwa, S. Samantaray, and G. Joos, "Development of rule-based classifiers for rapid stability assessment of wide-area post-disturbance records," *IEEE Trans. Power Syst.*, vol. 24, no. 1, pp. 258–270, Feb. 2009.

[15] I. Kamwa, "Catastrophe predictors from ensemble decision-tree learning of wide-area severity indices," *IEEE Trans. Smart Grid*, vol. 1, no. 2, pp. 144–158, Sep. 2010.

[16] M. He, J. Zhang, and V. Vittal, "A data mining framework for online dynamic security assessment: Decision trees, boosting, and complexity analysis," in *Proc. 2012 IEEE PES Innovative Smart Grid Technologies (ISGT)*, Jan. 2012, pp. 1–8.

[17] Y. Xu, Z. Y. Dong, J. H. Zhao, P. Zhang, and K. P. Wong, "A reliable intelligent system for real-time dynamic security assessment of power systems," *IEEE Trans. Power Syst.*, vol. 27, no. 3, pp. 1253–1263, Aug. 2012.

[18] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, ser. Springer Series in Statistics, 2nd ed. New York, NY, USA: Springer-Verlag, 2008.

[19] I. Genc, R. Diao, V. Vittal, S. Kolluri, and S. Mandal, "Decision tree-based preventive and corrective control applications for dynamic security enhancement in power systems," *IEEE Trans. Power Syst.*, vol. 25, no. 3, pp. 1611–1619, Aug. 2010.

[20] I. Kamwa, S. Samantaray, and G. Joos, "On the accuracy versus transparency trade-off of data-mining models for fast-response PMU-based catastrophe predictors," *IEEE Trans. Smart Grid*, vol. 3, no. 1, pp. 152–161, Mar. 2012.

[21] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*. London, U.K.: Chapman and Hall/CRC, 1984.

[22] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, pp. 119–139, 1997.

[23] A. Niculescu-mizil and R. Caruana, "Obtaining calibrated probabilities from boosting," in *Proc. 21st Conf. Uncertainty in Artificial Intelligence, AUAI Press*, 2005.

[24] R. Banfield, L. Hall, K. Bowyer, and W. Kegelmeyer, "A comparison of decision tree ensemble creation techniques," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 173–180, Jan. 2007.

[25] P. Utgoff, N. Berkman, and J. Clouse, "Decision tree induction based on efficient tree restructuring," *Mach. Learn.*, vol. 29, pp. 5–44, Oct. 1997.

[26] M. Box, D. Davies, and W. Swann, *Non-Linear Optimisation Techniques*. Edinburgh, U.K.: Oliver and Boyd, 1969.

[27] R. Schainker, G. Zhang, P. Hirsch, and C. Jing, "On-line dynamic stability analysis using distributed computing," in *Proc. 2008 IEEE Power and Energy Society General Meeting*, Jul. 2008, pp. 1–7.

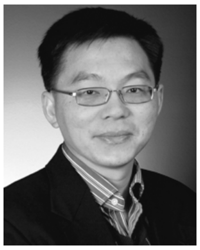
[28] M. Pai, *Energy Function Analysis for Power System Stability*. Boston, MA, USA: Kluwer, 1989.

[29] S. Chakrabarti and E. Kyriakides, "Optimal placement of phasor measurement units for power system observability," *IEEE Trans. Power Syst.*, vol. 23, no. 3, pp. 1433–1440, Aug. 2008.

[30] Powertech Labs, DSATools: Dynamic Security Assessment Software. [Online]. Available: http://www.dsatools.com.

[31] 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.

Miao He (S'08) received the B.S. degree from Nanjing University of Posts and Telecommunications, Nanjing, China, in 2005, and the M.S. degree from Tsinghua University, Beijing, China, in 2008. He is currently pursuing the Ph.D. degree at Arizona State University, Tempe, AZ, USA.



Junshan Zhang (F'12) received the Ph.D. degree from the School of Electrical and Computer Engineering at Purdue University, West Lafayette, IN, USA, in 2000.

He joined the Electrical Engineering Department at Arizona State University, Tempe, AZ, USA, in August 2000, where he has been a Professor since 2010. His research interests include communications networks, cyber-physical systems with applications to smart grid, stochastic modeling and analysis, and wireless communications.

Dr. Zhang is a recipient of the ONR Young Investigator Award in 2005 and the NSF CAREER award in 2003. He received the Outstanding Research Award from the IEEE Phoenix Section in 2003.

Vijay Vittal (S'78-F'97) received the B.E. degree in electrical engineering from the B.M.S. College of Engineering, Bangalore, India, in 1977, the M.Tech. degree from the India Institute of Technology, Kanpur, India, in 1979, and the Ph.D. degree from Iowa State University, Ames, IA, USA, in 1982.

He is the Ira A. Fulton Chair Professor in the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, USA. Currently, he is the director of the Power System Engineering Research Center (PSERC).

Dr. Vittal is a member of the National Academy of Engineering.