IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 2 February 2023, accepted 21 February 2023, date of publication 22 May 2023, date of current version 25 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3258680

RESEARCH ARTICLE

Assessing Bulk Power System Reliability by End-to-End Line Maintenance-Aware Learning

YONGLI ZHU, (Member, IEEE), AND CHANAN SINGH, (Life Fellow, IEEE)
Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77840, USA
Corresponding author: Yongli Zhu (yzhu16@vols.utk.edu)

ABSTRACT To overcome the slow-running drawback of the Monte Carlo simulation method for bulk power system reliability assessment, this paper develops an end-to-end machine learning approach to directly predict the targeted reliability index considering grid topology changes caused by emergent line maintenance. Three machine learning models, i.e., Support Vector Machine, Boosting Trees, and Graph Neural Network, are considered and compared. The grid topology information is embedded into the above models via two feature engineering schemes. Dataset creation and data preprocessing are also described. Then, two case studies with different experimental settings and prediction targets are performed on the IEEE RTS-79 system to inspect the proposed approach's adaptability. Results demonstrate the proposed approach's effectiveness and speed advantage. Finally, an analysis is presented regarding the Support Vector Machine's generalizability against the varying dataset size based on the empirical-risk theory from the machine learning community.

INDEX TERMS Boosting tree, maintenance-induced line outage, Monte Carlo simulation, reliability assessment, support vector machine, graph neural network.

I. INTRODUCTION

Reliability assessment (RA), a.k.a. generation adequacy assessment, is a common routine in power system planning and operation to evaluate the continuous load supply-ability. Modern reliability assessment studies can span large geographies to investigate the benefits of system interconnections. Thus, the impact of transmission networks must be considered in bulk power systems (BPS) reliability assessment. Analytic enumeration, Monte Carlo simulation (MCS), and hybrid methods are the three main approaches for BPS reliability assessment [1]. Analytical enumeration has merits in rigorousness and determinacy [2] but can become impractical due to the exponentially growing enumeration space when the system size increases. In contrast, the MCS method is flexible in considering dynamic system behaviors and human interactions [3] and is dimension-free of the total number of system states. However, the MCS method demands a sufficiently large number of simulations to reach convergence. The MCS method for reliability assessment has three major steps: a) sampling of the state (in non-sequential MCS) or event (in sequential MCS); b) testing of the state (i.e., to determine whether the sampled state is "reliable" or not; c) updating the corresponding statistics of the reliability indices, e.g., LOLP (loss of load probability) or EDNS (expected demand not served; also called "EPNS" in [1]). The hybrid method combines the above two methods, e.g., first applying the analytical state-space reduction techniques and then executing the MCS [4], [5].

In a real power grid, the system reliability indices need to be re-calculated when one or two lines are de-energized due to emergent maintenance needs (called *maintenance-induced line outage* in this paper). Though, for large-size systems, using the conventional workflow of MCS to evaluate *all* the line-outage cases can be time-consuming and laborious. One idea is to speed up the MCS itself, e.g., by using variance reduction techniques [6]. However, the MCS method still suffers the time cost issue due to the stochastic nature of the state-sampling step.

Meanwhile, data-driven techniques have become prevalent in the real-time operation and control of bulk power systems [7], [8], and studies have been presented regarding

The associate editor coordinating the review of this manuscript and approving it for publication was Zhiyi Li.

machine learning (ML) applications on reliability assessment. In [9], the author utilizes an improved Bayesian belief network model in an MCS-based unit commitment framework for reliability assessment. In [10], transfer learning techniques are adopted to adapt the reliability predictor for a varying system installed capacity. In [11], authors leverage the Genetic Algorithm (GA) for 1) intelligent state selection in the state-sampling step (by binary encoded GA) and 2) optimal load curtailment (by real encoded GA) in the state-testing step. In [12], a multi-label classifier is trained to output the reliability indices of all the buses.

However, the above-mentioned ML approaches have to be embedded in a conventional MCS framework. Thus, a large number of MCS runs are still needed in the subsequent stage to finally compute the reliability index. Per contra, the end-to-end machine learning approach can directly output the targeted reliability indices. For example, In [13], machine learning-based regression models are employed to map the power inverters' reliability indices to a high-level system reliability index. In [14], a brief study is conducted to predict the system LOLP by two shallow learning models. In [15], a natural language processing-based method is designed to forecast the outage duration of the distribution system.

In light of the above challenges, this paper proposes an end-to-end line maintenance-aware machine learning approach. The proposed approach can directly output the targeted reliability index considering maintenance-induced line outages up to a pre-defined order ($<=2$). The approach can help reduce the overall time cost compared to the conventional Monte Carlo workflow. More specifically, the main contributions of this paper are:

- Devise an end-to-end line maintenance-aware ML pipeline to *directly* predict a targeted reliability index. Two representative reliability indices of BPS are examined, viz. LOLP and EDNS.
- Develop systematic ways of feature engineering for *both* shallow and deep learning models to incorporate the maintenance-induced line outage information.
- Present an empirical risk analysis to help find a proper size of the training dataset in a *probabilistic* sense.

Note that: the "line maintenance" considered in this paper is due to the *unpredictable* (emergent) line maintenance needs in the system operation stage rather than *foreseeable* maintenance needs in the system planning stage.

In the following parts, Section II briefly introduces the basics of MCS for bulk power system reliability assessment. Section III illustrates the motivation and dataset creation of the devised end-to-end, line maintenance-aware ML pipeline. Section IV explains the basic principles and the proposed feature engineering schemes for three representative machine learning models: Support Vector Machine, Boosting Trees, and Graph Neural Network. Sections V and VI present case studies based on the IEEE RTS-79 system under two different settings. The final section summarizes this paper and provides future research directions.

II. MONTE CARLO SIMULATION FOR BPS RA

The MCS method for reliability assessment includes the non-sequential and sequential types based on different research needs [1], [6]. Either of them can be used in our approach. Since the MCS method per se is not the main focus of this paper, thus the non-sequential MCS method is adopted here, whose implementation is more straightforward during the dataset creation.

A. BASIC STEPS OF MONTE CARLO SIMULATION

The basic steps of non-sequential MCS for a system of p components are:

Step-1: Draw a system state $\boldsymbol{s} = (s_1, \ldots, s_p)$ based on each component's failure rate or probability distribution;

Step-2: Test the state by certain simple rules (e.g., comparing the total installed capacities with the total load) or by certain OPF (optimal power flow) models;

Step-3: Update the expectation and variance of the reliability index according to Eq. (1):

$$\mathbf{E}[g] = \frac{1}{N_s}\sum_{j=1}^{N_s} g(\mathbf{s}_j), \quad \mathbf{Var}[g] = \frac{1}{N_s}\sum_{j=1}^{N_s}(g(\mathbf{s}_j) - \mathbf{E}[g])^2 \tag{1}$$

where N_s is the total simulation runs. $\boldsymbol{s}_j$ is the system state sampled at the j-th run. g is a function mapping $\boldsymbol{s}_j$ to the targeted reliability index. $\mathbf{E}[\cdot]$ stands for the *expectation*. For instance, when g is in the form of Eq. (2), the expectation in Eq. (1) will represent the LOLP.

$$g(\mathbf{s}_j) = \begin{cases} 0, & \mathbf{s}_j \text{ is a reliable state} \\ 1, & \text{otherwise} \end{cases} \tag{2}$$

B. STATE TESTING VIA OPTIMAL POWER FLOW

Once a failure system state is drawn, the next step is to test this state by a tailored OPF model when the transmission network is considered [1], [6]. Compared to the DCOPF (direct current optimal power flow), stricter constraints (e.g., the limit of the bus voltage magnitude) can be considered via the ACOPF (alternative current optimal power flow). Hence, the state-testing step in this paper is all based on ACOPF (with customized objective functions or constraints). In fact, the ACOPF here tries to emulate the system operator's actions on a failure event.

C. STOPPING CRITERIA OF THE MCS PROCEDURE

As shown in Eq. (3), the "coefficient-of-variation" (denoted by β) is used for the stopping criteria in the MCS procedure for BPS RA [1]. When β is less than a threshold (e.g., 0.02), the MCS terminates, and the expectation in Eq. (1) will be taken as the final reliability index.

$$\mathbf{Var}[\mathbf{E}[g]] = \frac{1}{N_s}\mathbf{Var}[g], \quad \beta \triangleq \frac{\sqrt{\mathbf{Var}[\mathbf{E}[g]]}}{\mathbf{E}[g]} \tag{3}$$

Note that a smaller threshold can result in an unnecessarily long simulation time, hence increasing the overall time cost of dataset creation for machine learning. On the other hand, a larger threshold might lead to a less accurate estimation of the reliability index. Thus, a maximum iteration number is usually used together with a reasonably small threshold.

III. END-TO-END MACHINE LEARNING FOR BPS RA

A. MOTIVATION OF END-TO-END MACHINE LEARNING

The key point of most ML research mentioned in Section I is to *learn* the OPF behavior in the state-testing step: the output of the trained ML model is either the optimal load curtailment or a binary classifier about the sampled system state (e.g., indicating "reliable" or not). The outputs of the ML model are then utilized *inside* an MCS procedure at the *inference* stage. Thus, approaches in those research can be categorized as "ML-embedded Monte Carlo simulation." Its workflow is depicted in Fig. 1.

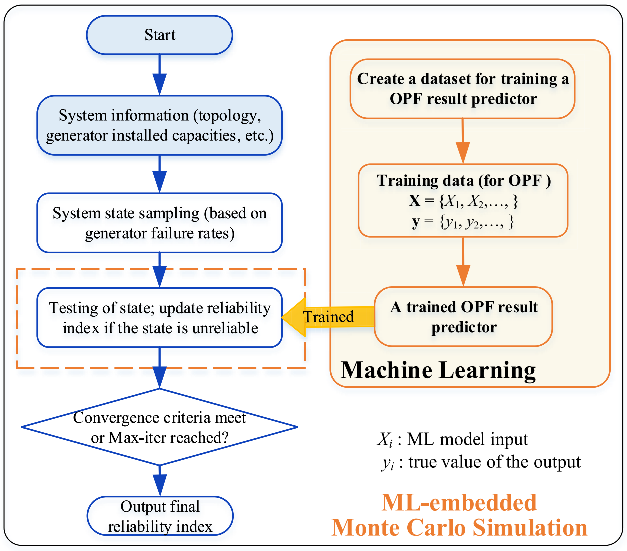
Start
System information (topology, generator installed capacities, etc.)
System state sampling (based on generator failure rates)
Testing of state; update reliability index if the state is unreliable
Convergence criteria meet or Max-iter reached?
Output final reliability index
Create a dataset for training a OPF result predictor
Training data (for OPF)
X = {X1, X2,…, }
y = {y1, y2,…, }
A trained OPF result predictor
Trained
Machine Learning
Xi : ML model input
yi : true value of the output
ML-embedded Monte Carlo Simulation

FIGURE 1. Reliability assessment by ML-embedded Monte Carlo simulation.

In Fig. 1, the model input $X_i = X_i(s_i)$ is the feature vector for the state s_i (e.g., generator installed capacity and line impedance). The model output y_i is the true value of a targeted reliability index, e.g., LOLP or EDNS. Compared to the conventional MCS method, this indirect-style workflow may help speed up the MCS. However, its limitations are:

- The MCS is still required in the outer loop and at the inference stage. Thus, the overall time cost is still large.
- When a classification model is adopted (e.g., for LOLP), the internal ML model suffers the notorious "class unbalance" issue: the number of negative samples (unreliable states, e.g., when power flow diverges or loss of load happens) is much rarer than that of the positive samples (reliable states). This limitation asks for extra handling, e.g., applying down-sampling techniques on the original dataset.

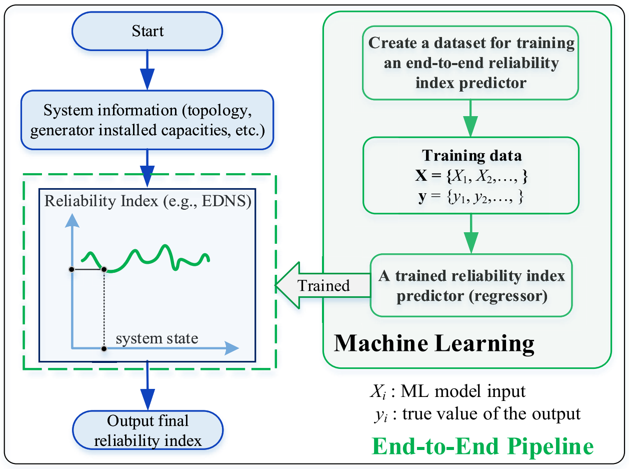
Start
System information (topology, generator installed capacities, etc.)
Reliability Index (e.g., EDNS)
system state
Output final reliability index
Create a dataset for training an end-to-end reliability index predictor
Training data
X = {X1, X2,…, }
y = {y1, y2,…, }
A trained reliability index predictor (regressor)
Trained
Machine Learning
Xi : ML model input
yi : true value of the output
End-to-End Pipeline

FIGURE 2. Reliability assessment by end-to-end machine learning.

On the contrary, an end-to-end machine learning pipeline is adopted in this paper, which can directly predict the targeted reliability index. Its workflow is illustrated in Fig. 2.

B. DATASET CREATION CONSIDERING LINE-MAINTENANCE

To generate the dataset for machine learning, the conventional MCS method is used. To consider the effect of maintenance-induced line outages, MCS procedures (cf Section II-A) are conducted for all the one-order cases (i.e., remove one line *before* the simulation starts) and two-order cases (i.e., simultaneously remove two lines *before* the simulation starts). In this paper, higher-order ($\geq$3) line-maintenance cases are not directly considered during the dataset creation since they are less common in real practice. Finally, the workflow for the dataset creation is shown in Fig. 3. Note that during the dataset creation:

- The above-mentioned "line-maintenance outages" are forced *before* starting the MCS procedure (rather than *inside* it).
- A grid-connectedness examination program [16] will be called for the above one- and two-order line-outage cases before the MCS procedure starts. All disconnected cases will be excluded from the final dataset for machine learning.
- Based on the forced-outage-rate of each unit, the (in-loop) outage events of units are considered up to three-order (i.e., the number of simultaneous unit outages in an internal iteration of MCS is up to three) because the higher-order unit outage events have lower chances to happen during the stochastic simulation.

IV. MACHINE LEARNING MODELS AND FEATURE ENGINEERING SCHEMES

Three iconic ML models are considered in this paper: Support Vector Machine (SVM), Boosting Trees (BT), and Graph Neural Network (GNN). SVM and BT are two well-known

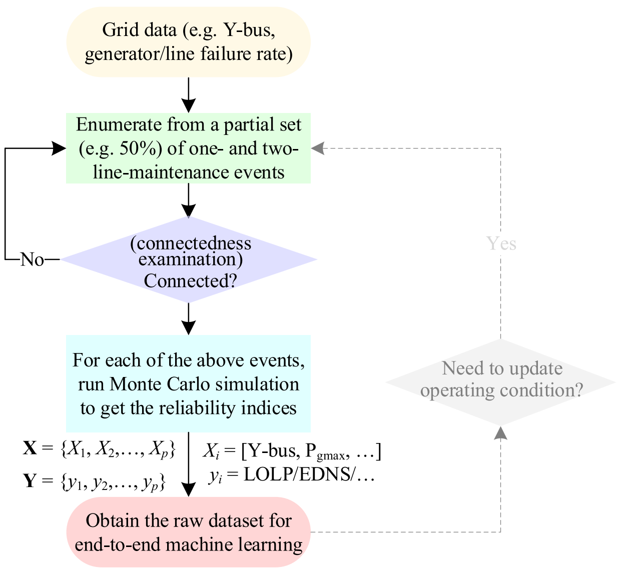
Grid data (e.g. Y-bus, generator/line failure rate)
Enumerate from a partial set (e.g. 50%) of one- and two-line-maintenance events
(connectedness examination) Connected?
No
Yes
For each of the above events, run Monte Carlo simulation to get the reliability indices
Need to update operating condition?
X = {X1, X2,…, Xp}
Y = {y1, y2,…, yp}
Xi = [Y-bus, Pgmax, …]
yi = LOLP/EDNS/…
Obtain the raw dataset for end-to-end machine learning

FIGURE 3. Dataset creation in the end-to-end line maintenance-aware ML pipeline for reliability assessment.

representatives of shallow (machine) learning models for regression [17], [18]. GNN, as a deep learning model, is suitable for graph data with topology variations [19], [20], [21].

A. ML MODEL-1: SUPPORT VECTOR MACHINE (SVM)

SVM is an ML model with good generalizability and a reasonable amount of hyperparameters, especially on small datasets [17]. The primal formulation of SVM for a regression task can be described by Eq. (4):

$$\begin{aligned} \min\ J(w,b,\xi_i,\xi_i^*) &= \frac{1}{2}||w||_2 + C\sum_{i=1}^{m}(\xi_i+\xi_i^*) \\ s.t.\ y_i - h(x_i) &\le \varepsilon + \xi_i, \quad i=1\ldots m \\ h(x_i) - y_i &\le \varepsilon + \xi_i^*, \quad i=1\ldots m \\ 0 \le \xi_i,\ 0 &\le \xi_i^* \quad i=1\ldots m \\ h(x) &\triangleq \langle w,x\rangle + b \end{aligned} \tag{4}$$

where: x_i and y_i are the input feature vector and the true value of the i-th data sample. w and b are the parameters to learn. m is the total number of data for learning. ξ_i and ξ_i^* are slack variables associated with each sample. ε is a small positive value (called *soft margin*) determined by hyperparameter optimization or specified by users. C is a positive constant controlling the penalty on samples outside the margin. A properly chosen C value can help avoid overfitting. In SVM's dual formulation, the concept of "kernel function" $K(x,\cdot)$ is useful, which maps the original features to higher dimensional space for better learning performance [18].

B. ML MODEL-2: BOOSTING TREES (BT)

The Boosting Trees (BT) aggregates a series of decision trees to reduce the risk of overfitting by a single tree. It utilizes a technique called *boosting* [17], i.e., concatenating *weak learners* (e.g., decision trees with merely one split) such that the newly obtained tree can improve the previous one's error. The algorithm below describes one version of the BT model, viz. *Least Square Boosting Tree*.

Algorithm: Least Square Boosting Tree

```
1 Input: $(x_i, y_i)$, $i = 1\ldots m$
2 $F_0(x) \leftarrow \bar{y}$ // using the average value of $y_i$ to initialize
3 for $k$=1 to $K$:
4     $\tilde{y}_i \leftarrow \bar{y}_i - F_{k-1}(x_i)$, $i = 1\ldots m$
5     $(\rho_k, \theta_k) \leftarrow \mathbf{argmin}_{\rho,\theta} \sum_{i=1}^{m} [\tilde{y}_i - \rho h(x_i;\theta)]^2$
6     $F_k(x) \leftarrow F_{k-1}(x) + \rho_k h(x;\theta_k)$
7 Output $F_K(x)$ as the final regressor
```

In the above pseudocode, F_k $(k = 0\ldots K)$ represents a series of tree regressors, and each is trained based on the residual of its predecessor. h stands for a simple decision tree. θ_m and ρ_m are the parameters to learn.

C. ML MODEL-3: GRAPH NEURAL NETWORK (GNN)

In the Graph Neural Network (GNN), the input signal can be either at the graph level or the vertex level. In the latter case, the representation of the graph can be obtained by certain *aggregation operators* on its vertex signals.

One classic architecture of GNN, viz. GCN (Graph Convolutional Network) [22], [23], [24], is defined based on the following *graph convolution* operation, as shown in Eq. (5):

$$\mathbf{X}' = \sigma\left(\beta(\mathbf{I}+\mathbf{L})\mathbf{X}\right) = \sigma\left(\beta\tilde{L}\mathbf{X}\right) \tag{5}$$

where $\mathbf{X}$ is the original graph signal. X' is the convolution result. σ is an activation function, e.g., $tanh(\cdot)$. β stands for the parameters to be learned. $\mathbf{I}$ is the identity matrix. $\mathbf{L}$ is the graph Laplacian (matrix) [22]. $\tilde{L}$ is the self-looped Laplacian, as defined by Eq. (6). $\deg(v_i)$ means the degree of vertex-i.

$$\tilde{L} \triangleq \mathrm{I}+\mathrm{L} = \begin{cases} 1, & if\ i=j \\ \dfrac{-1}{\sqrt{\deg(v_i)\deg(v_j)}}, & if\ e_{ij}\in E \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

The prediction task in this paper is at the graph level, as illustrated by Fig. 4, where the GNN takes a graph signal (assembled from its vertex signals) and finally generates the prediction for a targeted reliability index.

D. FEATURE ENGINEERING FOR SVM AND BT

To incorporate the impact of maintenance-induced line outages (i.e., "line-maintenance-aware"), the Y-bus matrix (i.e., its real part $\mathbf{G}$ and imagery part $\mathbf{B}$) can be leveraged. For each generator unit, the installed capacity (i.e., the power limit) $\mathbf{P_{gmax}}$ has to be used since it is directly related to the generation adequacy. Hence, each raw input data $X_{raw,i}$ can be initially constructed as:

$$X_{raw,i} = [\mathbf{G}, \mathbf{B}, \mathbf{P_d}, \mathbf{Q_d}, \mathbf{P_{gmax}}] \tag{7}$$

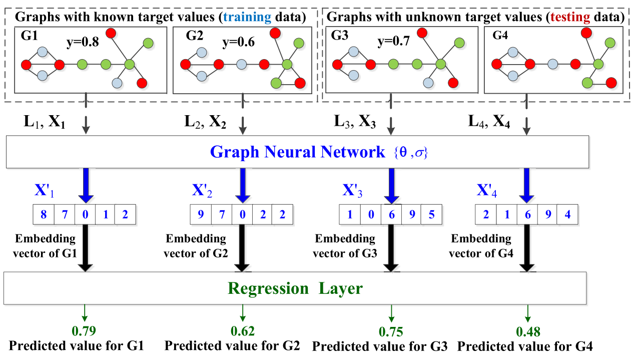

FIGURE 4. Illustration of using GNN for a graph-level regression task.

where $\mathbf{P_{gmax}}$ is mentioned above and zeroed for non-generator buses. $\mathbf{G}$ and $\mathbf{B}$ are n-by-n matrices (denote the total number of buses as n). $\mathbf{P_d}$ and $\mathbf{Q_d}$ are vectors of the active and reactive load (zeroed for the non-load buses). The purpose of appending the load vectors here is to prepare for future model-update (when the system loading condition is needed) by incremental training techniques.

The shape of each (raw) input data is n-by-$(2n + 3)$. However, this scheme has the following disadvantages:

- Memory space wasting due to the duplicated storage of symmetric elements in matrices $\mathbf{G}$ and $\mathbf{B}$.
- Slow training of the ML model due to the above redundant elements.

Hence, the following re-arrangement will be applied:

- Extracting the upper triangular part of $\mathbf{G}$ as a 1-dim column vector $\mathbf{vG}$; similarly, extracting $\mathbf{vB}$ from $\mathbf{B}$.
- Concatenating all those 1-dim vectors together.

In this way, about half of the space can be saved. This feature engineering scheme is depicted in Fig. 5. The new shape of each data sample becomes $n(n + 4)$-by-1, as shown in Eq. (8) (where ";" stands for the vertical concatenation):

$$X_i = [\mathbf{v_G}; \mathbf{v_B}; \mathbf{P_d}; \mathbf{Q_d}; \mathbf{P_{gmax}}]^{\mathrm{T}} \tag{8}$$

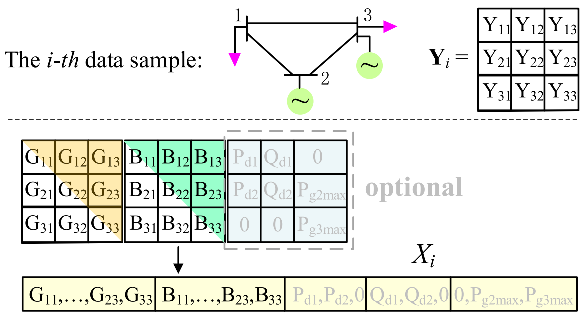
The i-th data sample:
1
2
3
Y_i =
Y11 Y12 Y13
Y21 Y22 Y23
Y31 Y32 Y33
G11 G12 G13 B11 B12 B13 Pd1 Qd1 0
G21 G22 G23 B21 B22 B23 Pd2 Qd2 Pg2max
G31 G32 G33 B31 B32 B33 0 0 Pg3max
optional
X_i
G11,…,G23,G33 B11,…,B23,B33 Pd1,Pd2,0 Qd1,Qd2,0 0,Pg2max,Pg3max

FIGURE 5. The proposed feature engineering scheme for SVM and BT.

E. FEATURE ENGINEERING FOR GNN

Similarly, the input of each data sample for the GNN is the same as Eq. (8). Note that the GNN also needs an edge index array and an edge weight array (if the graph is weighted) for each input, as shown in Eq. (9) and (10):

$$\mathbf{edge\_index}_i = [[f_1^i; t_1^i], \ldots, [f_l^i; t_l^i]] \in \mathbb{R}^{2-by-l} \tag{9}$$

$$\mathbf{edge\_weight}_i = [B_1^i, \ldots, B_l^i] \in \mathbb{R}^l \tag{10}$$

where. f_j^i and t_j^i are respectively the *from* and *to* bus (vertex) indices for the j-th line in the i-th data sample. l is the total number of lines in the i-th data sample. B_j^i is the susceptance value of the j-th line in the i-th data sample. Note that the GNN implementation in this paper is based on *PyG* [25], which requires the above two arrays also to contain the reversed edges for an undirected graph.

Finally, the true value of the targeted reliability index y_i and the obtained *feature embedding* vector will be sent to the "MSE (mean squared error) Loss" layer. This feature engineering scheme is depicted in Fig. 6.

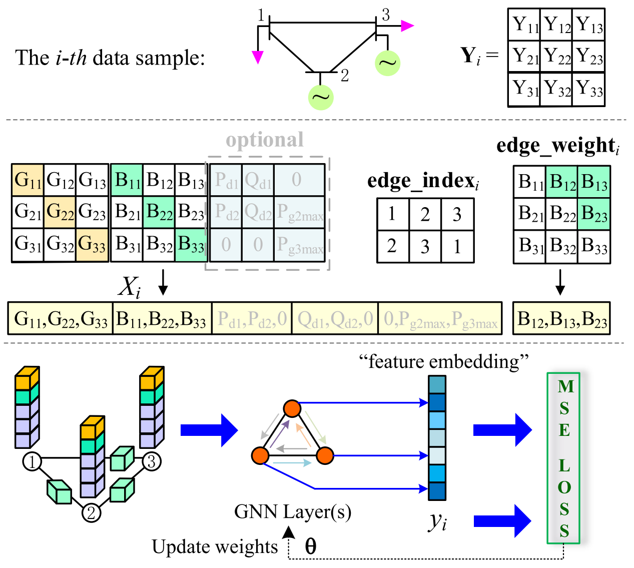

FIGURE 6. The proposed feature engineering scheme for GNN.

F. FEATURE SCALING

In this paper, the [0,1] scaling is applied to the input feature by Eq. (11):

$$x = (x - x_{\min})/(x_{\max} - x_{\min}) \tag{11}$$

where "x" is a feature vector of physical quantities of the same type. In this way, feature variables of different physical meanings are scaled separately to avoid numerical issues, i.e., vectors $\mathbf{v_G}$, $\mathbf{v_B}$, $\mathbf{P_d}$, $\mathbf{Q_d}$, and $\mathbf{P_{gmax}}$ are scaled using their respective maximum and minimum components.

V. CASE STUDY I

In this case study, the target to be predicted is the LOLP of the IEEE RTS-79 system. As shown in Fig. 7. The RTS-79

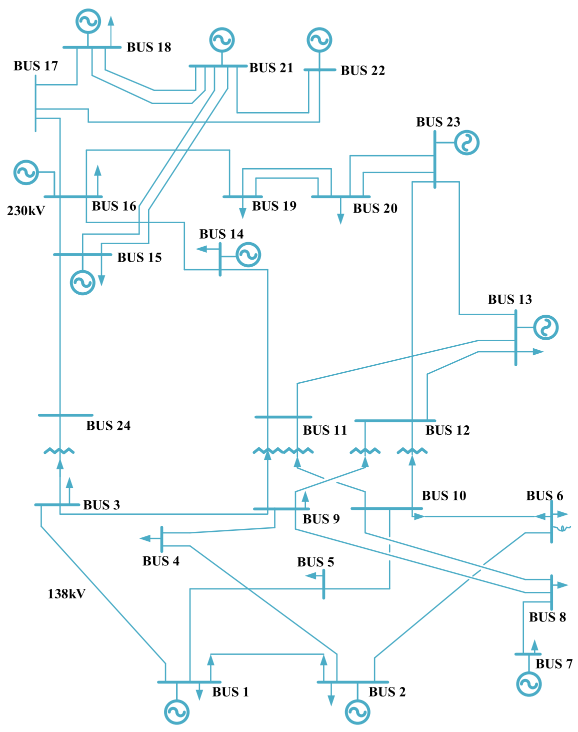

FIGURE 7. The single-line diagram of the IEEE RTS-79 system.

system has 24-bus, 32 generator units, and 38 lines. Three ML models, viz. SVM, BT, and GNN are respectively applied. The feature engineering for each ML model is described in the previous section. The dataset creation, model training, and testing are conducted on a computer with a 4.0Hz CPU and 16GB memory.

A. OPF MODEL FOR CREATING THE LOLP DATASET

The OPF model used in case study-1 is shown in Eq. (12):

$$\begin{aligned}
&\min_{V,Re(S^G)} \quad 0 \\
&s.t.\ \angle V_r = 0 && r: ref.bus\ no. \\
&\quad V_i^{\min} \le |V_i| \le V_i^{\max} && \forall i \in N \\
&\quad \theta_{ij}^{\min} \le \angle(V_i V_j^*) \le \theta_{ij}^{\max} && \forall (i,j) \in E \\
&\quad S_{ij} = Y_{ij}^*|V_i|^2 - Y_{ij}^* V_i V_j^* && \forall (i,j) \in E \\
&\quad |S_{ij}| \le S_{ij}^{\max} && \forall (i,j) \in E \\
&\quad S_i^{G,\min} \le |S_i^G| \le S_i^{G,\max} && \forall i \in N \\
&\quad S_i^G - S_i^D = \sum_{(i,j)\in E} S_{ij} && \forall i \in N \qquad (12)
\end{aligned}$$

where N and E are respectively the index sets of buses and lines. S_{ij} is the complex power flow on the branch $i-j$. S_i^G and S_i^D are respectively the complex power of the generator (if any) and load (if any) at the i-th bus. The meanings of other symbols are similar to those in a typical OPF formulation (cf. [6] for more details).

Note that:

TABLE 1. Basic performance (baseline: MCS).

	SVM	BT	GNN
MAE	0.0041	0.0045	0.0050
MAPE	4.53%	4.97%	5.35%
RMSE	0.0053	0.0053	0.0079

1) The objective function is preset to zero to obtain a *feasibility problem* since only the result of feasibility is useful in computing the LOLP. If this ACOPF model diverges, the tested state is deemed "unreliable," and the counter of the "LOLP event" will increase by one during the MCS procedure. The final LOLP is nothing but the counted value divided by the number of runs (cf. Eq. (1)).

2) In this case study, *before* the MCS starts, each case will be pre-checked by an initial ACOPF run. Any cases that fail this initial check will be excluded from the subsequent simulation and the final dataset. The reason is that: those cases will always fail all subsequent ACOPF runs *inside* the MCS; thus, their LOLP values are always equal to 1.0, eliminating the need for training.

Based on the steps in Section III, the total number of finally considered one-order cases is 34 after ruling out the divergent cases in the initial ACOPF check and the disconnected cases. Similarly, the total number of eventually considered two-order cases is 343. Thus 343 + 34 = 377 samples are collected. For each sample, one entire run of the MCS procedure takes about 200sec (with stopping criteria: $\beta \le 0.02$, maximum iterations = 5000).

B. BASIC PERFORMANCE OF THE ML MODELS

To inspect the basic performance of the proposed end-to-end predictor, a set of randomly picked 90% of the original data is used for training, and the left 10% is kept for testing. Three error metrics, MAE (mean-absolute-error), MAPE (mean-absolute-percentage-error), and RMSE (root-mean-square-error), are adopted for performance comparisons.

The linear kernel function is selected for the SVM based on trial and error. The maximum number of bottom-level trees for the BT is set to 100. For GNN, one GCN layer (hidden dimension = 16), one global-mean-pool layer [25], and one linear layer are used, with 0.002 as the learning rate and 64 as the batch size.

The error metrics are listed in Table 1. From the results, SVM is better than others. Plots of the predicted LOLPs versus the true values (from MCS) on the testing samples are shown in Fig. 8 (for better legibility, only SVM's predictions and the true values are depicted here).

C. IMPACT OF THE REDUCED TRAINING DATASET

If only partial data is available for training, the performance of the ML models may decline. To inspect this impact, three portions of the original dataset are used for training, i.e., 90%, 70%, and 50%. The respectively remaining data are retained for testing. The error metrics are listed in Table 2. As expected, the ML models' performance can deteriorate

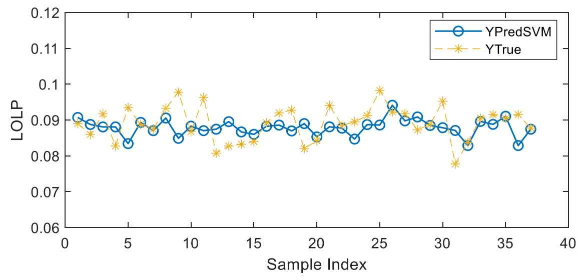

FIGURE 8. Comparison of SVM vs. Monte Carlo (truth) predictions on the testing dataset (90% data for training, 10% data for testing).

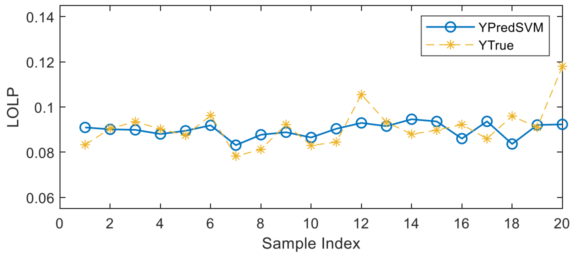

FIGURE 9. Comparison of SVM vs. Monte Carlo (truth) predictions on 20 unseen three-order samples (50% original data for training).

TABLE 2. Impact of the reduced training dataset.

		SVM	BT	GNN
MAE	90%	0.0041	0.0045	0.0050
	70%	0.0079	0.0086	0.0095
	50%	0.0084	0.0101	0.0091
MAPE	90%	4.53%	4.97%	5.35%
	70%	6.35%	7.74%	6.85%
	50%	6.23%	8.34%	6.45%
RMSE	90%	0.0053	0.0053	0.0079
	70%	0.0221	0.0205	0.0366
	50%	0.0310	0.0308	0.0347

TABLE 3. Generalizability on unseen three-order samples.

		SVM	BT	GNN
MAE	90%	0.0066	0.0092	0.0077
	70%	0.0067	0.0116	0.0077
	50%	0.0061	0.0096	0.0076
MAPE	90%	6.93%	9.97%	7.88%
	70%	7.03%	12.52%	7.90%
	50%	6.44%	10.13%	7.80%
RMSE	90%	0.0091	0.0146	0.0108
	70%	0.0096	0.0188	0.0109
	50%	0.0082	0.0132	0.0108

when training data size decreases; but in this case study, SVM still performs better than others in most metrics.

D. GENERALIZABILITY ON HETEROGENEOUS DATA SAMPLES

Recall that the original dataset *only* contains samples of one-order and two-order line outages. Thus, to inspect the generalizability of the trained ML model on unseen, *heterogeneous* data samples, a set of 20 three-order (maintenance-induced line outage) samples are randomly generated here for examination.

Models trained on three different portions (90%, 70%, and 50%) of the *original* training dataset are utilized here. The error metrics are shown in Table 3, where the best MAPE is achieved by SVM when 50% of the original training data is used. The comparison plot of SVM predictions versus the true values (from MCS) on those 20 three-order samples is shown in Fig. 9.

It is not astonishing that most metrics of all three methods have become larger because the testing samples here are not only *unseen* during the previous training process but also *heterogeneous*: no three-order line outage samples exist in the original dataset (created under one-order and two-order line outages). The SVM's error metrics are still smaller than others in this case study.

The result here implies that the trained end-to-end ML model still has certain *forecasting power* even on unseen, heterogeneous line-outage events.

E. COMPARISON OF TIME COSTS

Fig. 10. displays the time costs of the three ML models based on the experiments in Table 2. SVM has demonstrated an overall speed advantage. The training time costs of SVM are not obvious in the left subplot of Fig. 10 due to their much smaller values.

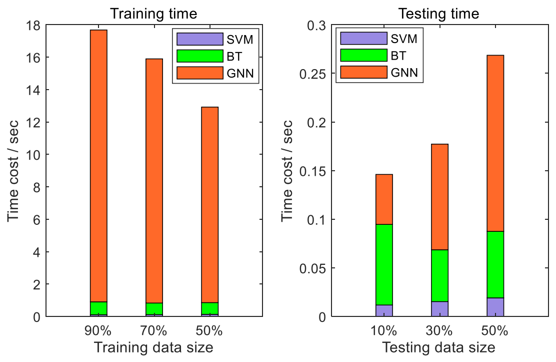

FIGURE 10. Time cost comparison for the three machine learning models.

In this case study, one entire run of the MCS procedure takes about 200sec for each sample. Hence, the conventional MCS workflow can take about 2 hours (377*0.1*200/3600 $\approx$ 2.09) on 10% of the original data, about 6 hours (377*0.3*200/3600 $\approx$ 6.28) on 30%, and so forth. In contrast, the end-to-end ML approach has more affordable time costs.

Moreover, if the assessment of three-order samples is desired, then the time-saving effect can be more remarkable. Hence, compared to the conventional MCS workflow, the end-to-end machine learning model can improve the efficiency of reliability assessment for systems under emergent line maintenance.

VI. CASE STUDY II

In this section, the RTS-79 system is still used, but a different setting is adopted to inspect the adaptability of the proposed approach on a different targeted reliability index, i.e., EDNS (expected demand not served, unit: MW). Note that in this case study:

- To predict the values of EDNS, a different OPF model is adopted, considering load curtailing actions.
- No ACOPF pre-check (cf. Section V-A) is needed since load curtailment will now be considered.

Here, a different OPF model is leveraged to calculate the (possible) unserved load demand during the Monte Carlo simulation for dataset creation. By trial and error, the linear kernel function is again used for the SVM, and the BT's maximum number of bottom-level trees is still set to 100. As for the GNN, two GCN layers (hidden dimension = 64) are used, with 0.02 as the learning rate and 64 as the batch size. All the ML models are re-trained based on the new settings and new dataset of this case study.

A. OPF MODEL FOR CREATING THE EDNS DATASET

The OPF model used in case study-2 is shown in Eq. (13):

$$\begin{aligned}
&\min_{V, Re(S^G), \Delta P_i^D} \sum_{i \in N_L} \Delta P_i^D \\
&s.t. \quad \angle V_r = 0 && r: ref.bus\ no. \\
&V_i^{\min} \le |V_i| \le V_i^{\max} && \forall i \in N \\
&\theta_{ij}^{\min} \le \angle(V_i V_j^*) \le \theta_{ij}^{\max} && \forall (i,j) \in E \\
&S_{ij} = Y_{ij}^* |V_i|^2 - Y_{ij}^* V_i V_j^* && \forall (i,j) \in E \\
&|S_{ij}| \le S_{ij}^{\max} && \forall (i,j) \in E \\
&S_i^{G,\min} \le |S_i^G| \le S_i^{G,\max} && \forall i \in N \\
&S_i^G - S_i^D = \sum_{(i,j)\in E} S_{ij} && \forall i \in N \backslash N_L \\
&S_i^G - S_i^D + \Delta P_i^D = \sum_{(i,j)\in E} S_{ij} && \forall i \in N_L \qquad (13)
\end{aligned}$$

where ΔP_i^D is the extra decision variable, i.e., the load curtailment, and N_L is the index sets of load buses. The meanings of other symbols are similar to Section V. In this OPF model, the goal is to minimize the possible load curtailment, which is equivalent to the possible load loss in one internal iteration of the MCS procedure, and the final averaged value (expectation) will be used as the EDNS.

Based on the steps in Section III, the total number of finally considered one-order cases is 37 after removing the divergent cases in the initial ACOPF check and the disconnected cases. Similarly, the total number of considered two-order cases is 659. 37 + 659 = 696 data samples are collected in total. For each data sample, one entire run of the MCS procedure takes about 297sec (stopping criteria: $\beta \le 0.02$, maximum iterations = 100000).

TABLE 4. Basic performances (baseline: MCS).

	SVM	BT	GNN
MAE	0.9429	0.9116	1.4723
MAPE	2.60%	2.65%	5.19%
RMSE	0.7400	0.7556	1.8217

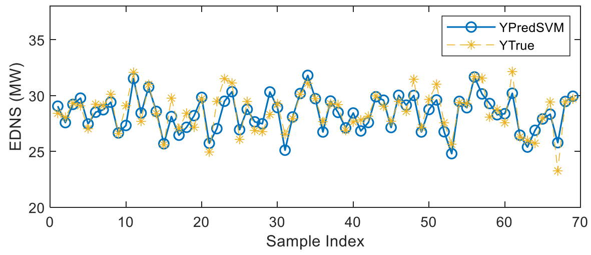

FIGURE 11. Comparison of SVM vs. Monte Carlo (truth) predictions on the testing dataset (90% data for training, 10% data for testing).

TABLE 5. Impact of the reduced training dataset.

		SVM	BT	GNN
MAE	90%	0.9429	0.9116	1.4723
	70%	0.7344	0.7420	1.4299
	50%	0.7438	0.8179	1.4550
MAPE	90%	2.60%	2.65%	5.19%
	70%	2.58%	2.60%	5.09%
	50%	2.62%	2.89%	5.15%
RMSE	90%	0.7400	0.7556	1.8217
	70%	1.0162	1.0559	1.7626
	50%	0.9814	1.1059	1.7864

B. BASIC PERFORMANCE OF THE ML MODELS

Similar to Section V-B, a set of randomly picked 90% of the data is used for training, and the left 10% is retained for testing. The error metrics are listed in Table 4. From the results, SVM performs best. The plots of the predicted EDNS versus the true values (from MCS) on the testing samples are shown in Fig. 11 (for better legibility, only SVM's predictions and the true values are depicted here).

C. IMPACT OF THE REDUCED TRAINING DATASET

Similar to Section V-C, three portions of the training dataset are inspected, i.e., 90%, 70%, and 50%. The respectively remaining data are retained for testing. The error metrics are listed in Table 5. Again, SVM still performs better than others in most metrics in this case study.

D. GENERALIZABILITY ON HETEROGENEOUS DATA SAMPLES

Similar to Section V-D, 20 three-order line outage samples are randomly generated for examination. Models trained on three different portions (90%, 70%, and 50%) of the *original* training dataset are utilized here. The three error metrics are shown in Table 6, where the best MAPE is achieved by SVM when 70% of the original training data is used. The comparison plot of SVM predictions versus the true values (from MCS) on those 20 three-order samples is shown in

TABLE 6. Generalizability on unseen three-order samples.

		SVM	BT	GNN
MAE	90%	1.0222	1.1364	1.3578
	70%	0.9072	1.0230	1.3914
	50%	0.9432	1.1779	1.3518
MAPE	90%	3.73%	4.16%	4.96%
	70%	3.27%	3.72%	5.11%
	50%	3.42%	4.31%	4.93%
RMSE	90%	1.3080	1.4753	1.7535
	70%	1.1011	1.3843	1.8022
	50%	1.2058	1.5340	1.7437

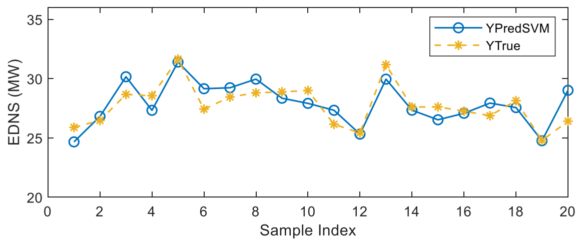

FIGURE 12. Comparison of SVM vs. Monte Carlo (truth) on 20 unseen three-order samples (70% original data for training).

Fig. 12. Here, the SVM's error metrics are still smaller than the other two ML models.

E. COMPARISON OF TIME COSTS

In this case study, one entire run of the MCS procedure takes about 297sec for each sample. Hence, if using the conventional MCS workflow, it will take about 6 hours (696*0.1*297/3600 $\approx$ 5.74) on 10% of the original data, about 17 hours (696*0.3*297/3600 $\approx$ 17.22) on 30%, and so on. Again, if using the end-to-end ML approach, both training and testing time costs are more affordable.

VII. A THEORETICAL ANALYSIS OF THE GENERALIZABILITY OF SVM

Since SVM performs best in the previous case studies of this paper, here, a brief generalizability analysis is presented for SVM based on the concept of *empirical risk* from the machine learning theory [18]. Denote the sample set as $S = \{z_1, z_2, \ldots z_m\}$, $z_i = (x_i, y_i) \in \mathbf{X} \times \mathbf{Y}$ (here $\mathbf{X}$ and $\mathbf{Y}$ stand for the domains of input and output); a "hypothesis function" (need to learn) as $h \in$H (H is a specified family of functions, e.g., the linear function in Eq. (4)); the loss function as $L_z = L(h(x), y)$; the latent probability distribution D (typically unknown) from which the samples $\{z_i\}$ are drawn.

Definition 1: the *generalization risk* is defined by

$$R(h) = \mathop{\mathbf{E}}_{z\sim D}[L_z(h)] = \mathop{\mathbf{E}}_{(x,y)\sim D}[L(h(x), y)] \tag{14}$$

Definition 2: the *empirical risk* is defined by

$$\widehat{R}_S(h) = \frac{1}{m}\sum_{i=1}^{m} L_{z_i}(h) = \frac{1}{m}\sum_{i=1}^{m} L(h(x_i), y_i) \tag{15}$$

Then, by Corollary 14.5 in [18], for any $\delta > 0$, the following inequality holds with probability at least $1-\delta$:

$$R(h_S) \leq \widehat{R}_S(h_S) + \underbrace{\frac{r^2}{m\lambda} + (\frac{2r^2}{\lambda} + M)\sqrt{\frac{\log(1/\delta)}{2m}}}_{W} \tag{16}$$

where:

- h_S is the hypothesis function returned by training the SVM model on the given dataset S;

- r is an upper bound for the kernel function $K(x, x)$ used in the SVM regressor (e.g., when using the linear kernel, $K(x, x) = ||x||^2$) and x stands for the input feature vector;

- λ is the coefficient of the penalty term in the SVM, i.e., the parameter C in Eq. (4);

- W is called the "incremental bound of the empirical risk."

- M is an upper bound for the specific loss function L in the SVM regressor, e.g., the loss function used in this paper's SVM is shown in Eq. (17), where $\tilde{y}_i (= h_S(x_i))$ is the predicted value by SVM; y_i is the true value. ε is the soft-margin parameter (cf. Eq. (4)).

$$L(\tilde{y}_i, y_i) := \begin{cases} 0, & \text{if} |\tilde{y}_i - y_i| \leq \varepsilon \\ |\tilde{y}_i - y_i| - \varepsilon, & \text{otherwise} \end{cases} \tag{17}$$

It should be noted that the right-hand side of Eq. (16) can be much larger than the left-hand side (generalization risk). Nevertheless, it is still useful for theoretical analysis, e.g., to evaluate the algorithmic stability and "fitting power" of the regressor (SVM). Besides, it can be inferred from (16) that W (the lower, the better, in terms of "generalizability") is *not always* linearly dependent on the data size of m: a larger dataset may also lead to bigger M and smaller λ.

Inspired by the above observation, an empirical method is proposed in this paper to *approximately* find an acceptable "minimum" size of the data set. Take the example of case study-2. Suppose 50% of the overall sample space has been initially generated and used in training. Then, the parameters of Eq. (16) can be estimated as follows (based on Table 5):

1) Estimate r based on the feature engineering scheme (note that r does not necessarily rely on the training data size m). In case study-2, it can be estimated by Eq. (18):

$$||X_i|| = ||[\mathbf{v_G}; \mathbf{v_B}; \mathbf{P_d}; \mathbf{Q_d}; \mathbf{P_{gmax}}]^{\mathrm{T}}|| \leq 13.22 \Rightarrow r = 13.22 \tag{18}$$

2) After training, record the values of the MAE, λ, and ε, i.e.,

$$\lambda = 1.8367, \quad \varepsilon = 0.01093, MAE = 0.7438 \tag{19}$$

3) Then, estimate the parameter M as:

$$M \approx MAE - \varepsilon = 0.7329 \tag{20}$$

4) Take a desired value of δ (e.g., $\delta = 0.1$, which corresponds to $1 - 0.1 = 90\%$ probability in the above Corollary). Plugging the above parameter values in that W term of

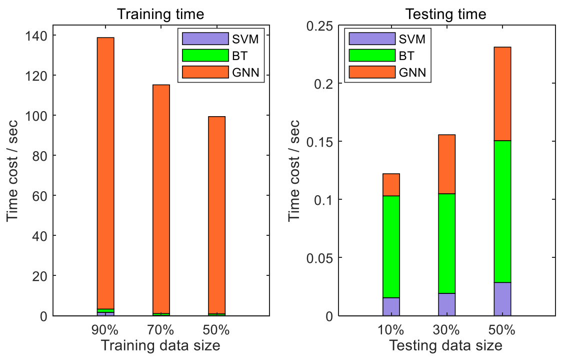

FIGURE 13. Time cost comparison for the three machine learning models.

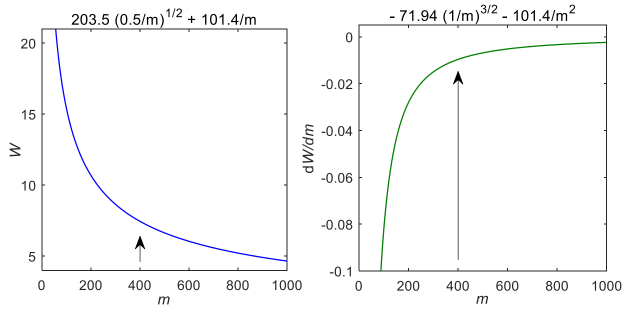

FIGURE 14. The incremental bound of the empirical risk (left) and its first-order derivative (right) vs. the training data size.

Eq. (16) to obtain a single-variable function of m (the training data size).

5) Lastly, draw the function curve of the term W and its first derivative versus m, as shown in Fig. 14. Empirically pick a proper size on which W starts to become small, or the first order derivative of W starts to approach its maximum.

From the curve, it can be observed that the decreasing speed of the incremental risk term W starts to become small when the size $m \geq 400$. Thus, empirically, the training process can be stopped safely at this point (in a probabilistic sense) since continued training on extra data may bring relatively less improvement to the ML model's performance.

VIII. CONCLUSION AND FUTURE WORK

In this paper, the devised end-to-end, line maintenance-aware machine learning pipeline shows acceptable accuracy and generalizability on partial data and heterogenous inputs when using the support vector machine. Its advantage in time-saving over the conventional MCS workflow is also demonstrated. Future work is 1) considering transfer learning for further time-saving and 2) experimenting other machine learning models for further improvement in accuracy.

REFERENCES

[1] C. Singh, P. Jirutitijaroen, and J. Mitra, *Electric Power Grid Reliability Evaluation: Models and Methods*. Piscataway, NJ, USA: Wiley, 2018.

[2] K. H. Youssef, "Microgrid reliability considering directional protection failure and optimal load shedding," *IEEE Trans. Smart Grid*, vol. 13, no. 2, pp. 877–887, Mar. 2022, doi: 10.1109/TSG.2021.3124929.

[3] B. Johnson, V. Chalishazar, E. Cotilla-Sanchez, and T. K. A. Brekken, "A Monte Carlo methodology for earthquake impact analysis on the electrical grid," *Electr. Power Syst. Res.*, vol. 184, Jul. 2020, Art. no. 106332, doi: 10.1016/j.epsr.2020.106332.

[4] W. Liu, D. Guo, Y. Xu, R. Cheng, Z. Wang, and Y. Li, "Reliability assessment of power systems with photovoltaic power stations based on intelligent state space reduction and pseudo-sequential Monte Carlo simulation," *Energies*, vol. 11, no. 6, p. 1431, Jun. 2018.

[5] S. Hou, P. Zhang, K. Hou, W. Zhang, Z. Shen, Q. Xiao, and Y. Lei, "Contingency set partition-based impact transfer approach for the reliability assessment of composite generation and transmission systems," *Int. J. Electr. Power Energy Syst.*, vol. 122, Nov. 2020, Art. no. 106130, doi: 10.1016/j.ijepes.2020.106130.

[6] R. Billinton and W. Li, *Reliability Assessment of Electric Power Systems Using Monte Carlo Methods*. Boston, MA, USA: Springer, 1994.

[7] R. D. Quint, "Data-driven engineering: The reliability and resilience of the North American bulk power system [technology leaders]," *IEEE Electrific. Mag.*, vol. 9, no. 1, pp. 5–9, Mar. 2021, doi: 10.1109/MELE.2020.3047162.

[8] T. Dimitrovska, U. Rudez, and R. Mihalic, "Indirect power-system contingency screening for real-time applications based on PCA," *IEEE Trans. Power Syst.*, vol. 33, no. 1, pp. 1080–1081, Jan. 2018, doi: 10.1109/TPWRS.2017.2691556.

[9] T. Lin and C. Shang, "Reliability evaluation on a joint machine learning and optimization framework," *IEEE Trans. Power Syst.*, vol. 36, no. 1, pp. 49–57, Jan. 2021.

[10] D. Urgun and C. Singh, "Composite system reliability analysis using deep learning enhanced by transfer learning," in *Proc. Int. Conf. Probabilistic Methods Appl. Power Syst. (PMAPS)*, Aug. 2020, pp. 1–6.

[11] N. Samaan, "Reliability assessment of electric power systems using genetic algorithms," Ph.D. dissertation, Dept. Elect. Eng., Texas A&M Univ., College Station, TX, USA, 2004.

[12] D. Urgun and C. Singh, "Power system reliability evaluation using Monte Carlo simulation and multi label classifier," in *Proc. 20th Nat. Power Syst. Conf. (NPSC)*, Dec. 2018, pp. 1–6.

[13] B. Zhang, M. Wang, and W. Su, "Reliability analysis of power systems integrated with high-penetration of power converters," *IEEE Trans. Power Syst.*, vol. 36, no. 3, pp. 1998–2009, May 2021.

[14] A. Jaech, B. Zhang, M. Ostendorf, and D. S. Kirschen, "Real-time prediction of the duration of distribution system outages," *IEEE Trans. Power Syst.*, vol. 34, no. 1, pp. 773–781, Jan. 2019.

[15] Y. Zhu and C. Singh, "End-to-end topology-aware machine learning for power system reliability assessment," in *Proc. 17th Int. Conf. Probabilistic Methods Appl. Power Syst. (PMAPS)*, Jun. 2022, pp. 1–6.

[16] Y. Zhu, L. Shi, R. Dai, and G. Liu, "Fast grid splitting detection for N-1 contingency analysis by graph computing," in *Proc. IEEE Innov. Smart Grid Technol. Asia (ISGT Asia)*, May 2019, pp. 673–677, doi: 10.1109/ISGT-Asia.2019.8880879.

[17] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*, 2nd ed. New York, NY, USA: Springer-Verlag, 2017.

[18] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.

[19] C. Kim, K. Kim, P. Balaprakash, and M. Anitescu, "Graph convolutional neural networks for optimal load shedding under line contingency," in *Proc. IEEE Power Energy Soc. Gen. Meeting (PESGM)*, Aug. 2019, pp. 1–5, doi: 10.1109/PESGM40551.2019.8973468.

[20] G. Wang, Z. Zhang, Z. Bian, and Z. Xu, "A short-term voltage stability online prediction method based on graph convolutional networks and long short-term memory networks," *Int. J. Electr. Power Energy Syst.*, vol. 127, May 2021, Art. no. 106647, doi: 10.1016/j.ijepes.2020.106647.

[21] Y. Zhu and C. Singh, "Topology-aware reliability assessment by graph neural networks," in *Proc. IEEE Kansas Power Energy Conf. (KPEC)*, Apr. 2022, pp. 1–6.

[22] T. N. Kipf and M. Welling. (2017). *Semi-Supervised Classification With Graph Convolutional Networks*. [Online]. Available: https://openreview.net/forum?id=SJU4ayYgl

[23] Y. Hechtlinger, P. Chakravarti, and J. Qin, "A generalization of convolutional neural networks to graph-structured data," *CoRR*, vol. abs/1704.08165, pp. 1–14, Apr. 2017. [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1704.html#HechtlingerCQ17

[24] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. NIPS*, 2017, pp. 1024–1034.

[25] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch geometric," in *Proc. ICLR Workshop Represent. Learn. Graphs Manifolds*, 2019, pp. 1–9.

YONGLI ZHU (Member, IEEE) received the B.S. degree from the Huazhong University of Science and Technology, in 2009, and the Ph.D. degree from The University of Tennessee, Knoxville. He has been worked as a Postdoctoral Researcher at Texas A&M University, College Station, TX, USA, since 2021. His research interests include power system reliability, power system stability, and the applications of artificial intelligence to power systems.

CHANAN SINGH (Life Fellow, IEEE) received the D.Sc. degree from the University of Saskatchewan, Saskatoon, SK, Canada, in 1997. He is a University Distinguished Professor, a Regents Professor, and an Irma Runyon Chair Professor with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, USA. He was the Department Head of Electrical and Computer Engineering at Texas A&M University, from 1997 to 2005, and an Interim Head, from 2012 to 2015. He was also a Program Director with the National Science Foundation of USA and a Guest Professor with Tsinghua University. He has authored/coauthored more than 400 technical articles and four books and has contributed to several books. He has also developed and contributed to computer programs for reliability evaluation used by the power industry. He has consulted with many major corporations and given short courses nationally and internationally. His research interests include the foundational developments and applications of probabilistic methods for planning and operation of electric power grid.

He is a member of the U.S. National Academy of Engineering and a fellow of the Indian National Academy of Engineering. He is also a fellow of the Chinese Society of Electrical Engineering. He was a recipient of the 1998 Outstanding Power Engineering Educator Award by the IEEE Power Engineering Society. He was recognized with the Merit Award by the PMAPS International Society for lifelong achievements. He was an inaugural recipient of the IEEE-PES Roy Billinton Power System Reliability Award and the IEEE-PES Lifetime Achievement Award.