Machine Learning Assisted Stochastic Unit Commitment during Hurricanes with Predictable Line Outages

Farshad Mohammadi[1], *Student Member, IEEE*, Mostafa Sahraei-Ardakani[1], *Member, IEEE,* Dimitris N. Trakas[2], *IEEE, Member,* and Nikos D. Hatziargyriou[2], *Fellow, IEEE*

Abstract— Stochastic unit commitment is an efficient method for grid operation in the presence of significant uncertainties. An example is an operation during a predicted hurricane with uncertain line out-ages. However, the solution quality comes at the cost of substantial computational burden, which makes its adoption challenging. This paper evaluates some possible ways that machine learning can be used to reduce this computational burden. First, a set of feasibility studies is conducted. Results suggest that using machine learning as an assistant to the stochastic unit commitment solver is more advantageous than using it as a standalone solver. In particular, the machine learning model is trained to facilitate solving the problem by determining the unnecessary constraints that can be removed from the original problem without affecting the final accuracy. The variables that can be used as input features/predictors or outputs for the machine learning model are determined through feasibility studies. Then, an algorithm to train and utilize a machine learning model is proposed. The method is tested on a 500-bus synthetic South Carolina system. Various test cases show an average reduction in solution time by more than 90% by using the trained machine learning model to assist the stochastic unit commitment solver.

***Index Terms*— Large-scale systems, load shedding, machine learning, ML-assisted power system operation, power outage, power system resiliency, preventive operation, stochastic unit commitment, severe weather, transmission line outage.**

NOMENCLATURE

A. Sets

g	Index of the generator, $g \in G$
n	Index of the bus, $n \in N$
k	Index of the transmission line and transformer, $k \in K$
m	Index of the monitored transmission line, $m \in M$
s	Index of the scenario, $s \in S$
o	Index of the line outage, $o \in O$
t	Index of the time interval, $t \in T$
frm	Set of starting bus of lines
to	Set of ending bus of lines

B. Parameters

c	Cost of generation
c^{NL}	No-load cost for generator
c^{SU}	Start-up cost for generator
c^{SD}	Shut-down cost for generator
c^{lsh}	Load shedding cost (penalty)
π	Scenario possibility
PG^{max}	Maximum generation power by generator
PG^{min}	Minimum generation power by generator
F^{max}	Maximum thermal capacity of the line
$\mathbf{PTDF}$	Power transfer distribution factor matrix

C. Variables

$\mathbf{F}$	Line flow vector
$\mathbf{FC}$	Flow canceling transactions vector
$\mathbf{P}$	Net nodal injected power vector
PG	Generated power of a generator
P^d	Power demand at bus
P^{lsh}	Load shedding
u	Unit commitment binary variable
v	Generator start-up binary variable
x	Generator Shut down binary variable

I. INTRODUCTION

Unit commitment (UC) is widely used in power system operation for generation scheduling, risk analysis, and planning in power systems [1], [2]. Simplified unit commitment formulations, based on DC power flow, take advantage of various techniques to solve the problem [3]. While UC, even in its simplest form, is difficult to solve, it becomes more challenging when additional uncertainties are considered. This is the case when a severe weather condition is predicted to hit the grid and cause damage to the transmission network.

Stochastic Unit Commitment (SUC) is among the most efficient methods for addressing uncertainties in the UC problem [4], [5]. Solving the SUC problem with explicit modeling of uncertainties, even with simplifying techniques, is challenging. Over the years, researchers and system operators have made progress in SUC problem modeling and developing solution algorithms [6]. Scenario-based SUC is among the most common approaches for modeling uncertainties within a stochastic framework [7]. In scenario representation, a set of scenarios over the uncertain future are defined and used to model the stochastic nature of uncertainties as a set of discrete futures. To

[1] Farshad Mohammadi and Mostafa Sahraei-Ardakani are with the Department of Electrical and Computer Engineering, University of Utah, Salt Lake City, UT, USA (e-mails: farshad.mohammadi@utah.edu, mostafa.ardakani@utah.edu.)

[2] D. N. Trakas and N. D. Hatziargyriou are with the Electrical and Computer Engineering, National Technical University of Athens, Greece (e-mails: dtrakas@power.ece.ntua.gr, nh@power.ece.ntua.gr).

This research was funded by the NSF ECCS grant # 1839833.

achieve quality results, scenario generation, reduction, and aggregation must be properly performed. Generally, the higher the number of scenarios, the better the quality of the solution; this, however, comes at the cost of heavier computation [8]. Scenario-based SUC has the primary advantage of explicitly modeling the uncertainties into the problem, thus offering a reliable solution [7]. However, its main disadvantage that limits its applications is the demanding calculation time.

SUC based on scenario representation is also linked to forecasting [9]. A vast literature can be found on using scenario-based SUC to represent forecasting and the related uncertainties in the network. As examples, [9]–[12] address the uncertainties related to the renewable generation forecast, while [13], [14] focus on demand forecast uncertainties. While modeling new uncertainties into the SUC adds to the complexity of the problem, studies such as [15] claim that by using high-performance computing systems and utilizing parallelization, it is possible to solve the problem within an acceptable time for online and offline applications with uncertainties related to generation, transmission failures and demand. However, given the need for modern power systems to consider more and more uncertainties, it is desirable to develop SUC computational approaches further. The modern network also requires multiple sources of uncertainties to be modeled to achieve an efficient solution [16]–[18]. Enhancement of modeling the SUC problem is necessary when the scope of the target uncertainties is beyond the operation of the system under normal conditions. An example of such an application is when a severe weather event, such as a hurricane, is predicted to impact the network [8], [19]. Hurricanes can cause tens or even hundreds of transmission outages over the course of their impact, often within a few hours. Considering possible outages, the optimal scheduling of generators that minimizes load shedding and enhances power system resilience can be beneficial [15], [20]–[22]. Although these outages can be predicted in advance, the predictions include uncertainties [23]. While scenario selection for preventive stochastic unit commitment during hurricanes is difficult, solving the problem within a sufficient time can be challenging as well.

Despite having different well-known techniques for scenario generation, reduction, and aggregation with applications in power systems such as [7], [24]–[26], they may not be fit for extreme cases, such as operation during a hurricane. This is due to the probabilistic nature of line failures and that the set of possible network topologies might be extremely large [8]. An efficient method to generate scenarios, called multidimensional scenario selection (MDSS), is introduced in [8], where multiple aspects of information regarding each uncertainty are used to produce the preferred number of scenarios. In [27], [28], an enhanced formulation capable of handling multiple line outages is introduced to model the problem as a preventive SUC problem. While a combination of the MDSS and the formulation introduced in [27] and [28] delivers a high-quality solution, the overall required time can still be very long for large networks. In this paper, we evaluate the feasibility of using Machine Learning (ML) algorithms to reduce the calculation time of the SUC problem.

ML techniques have been successfully used for more than a decade to solve security-constrained UC problems in order to determine dynamic security constraints, e.g. [29], or to solve dynamic security problems, such as in [30], [31]. Reference [32] reviews the applications of ML in energy system reliability management, including assistance in solving the SUC problem. Ref. [32] mentions eight studies, each proposing a model to use ML algorithms in energy systems. References [33] and [34] claim that it is possible to estimate the final solution when planned outages are evaluated by utilizing ML with long offline training time. However, with a lower quality of the final solution in SUC In [35], authors suggest that instead of solving the complete form of transmission-constrained UC, it is possible to use an ML model to learn about congestions in the network and reduce the unnecessary constraints from the problem. In the mentioned study, it is assumed that the network topology remains unchanged, and the congestion patterns do not change over the study horizon of their data-driven method. In [36], the same authors of [33] and [34] model the UC problem as a Markov decision process and solve it by approximating the results through reinforcement learning. Finally, [37] and [38] model the UC as a multi-objective optimization problem and evaluate system reliability.

This paper is different since it investigates how ML can improve solution time without sacrificing accuracy and without the need for historical data of network operation. Moreover, as this paper's main application is the SUC when severe weather is predicted to hit the grid, unlike most of the literature, the network topology is not assumed constant. Additionally, this paper investigates the sensitivity of a feasible solution to the error, when an imperfect ML model is trained and used.

The general idea can be summarized as follows: The objective of the paper is to reduce the calculation time when solving the SUC problem with multiple probabilistically predicted line outages by using ML methods. Section II reviews the original SUC problem and explains the use of ML to reveal the challenges of the original problem and the capabilities and limitations of ML. Considering that there are different possible ways that ML can be utilized, a feasibility study is performed in Section III to determine how ML can help to solve the problem faster. In Section IV, a suitable algorithm is proposed to train the ML model(s) and use the trained model(s) to solve the SUC. To demonstrate the quality and efficiency of the proposed method, different numerical simulations are conducted in Section V, and the results are discussed and evaluated, confirming the supremacy of the proposed method. Finally, Section VI concludes the paper.

II. Background

This section explains the original SUC problem, the challenges of solving the model, and the potentials for using ML to facilitate its solution.

A. Original SUC Problem

The goal of SUC is to minimize the expected value of the objective function, usually operation cost, subject to physical and reliability constraints. With high levels of uncertainties, the objective function should include not only the generation costs but also penalized load shedding. The failure of many transmission lines during hurricanes, and hence possible disconnection of operating generation unit(s) or load(s), may lead to inevitable

load shedding. The objective function is defined as:

$$Minimize \sum_{s\in S}\{\pi_{(s)} \sum_{t\in T}[\sum_{g\in G}(c_{(g)}PG_{(s,g,t)} + c_{(g)}^{NL}u_{(s,g,t)} + c_{(g)}^{SU}v_{(s,g,t)} + c_{(g)}^{SD}x_{(s,g,t)}) + \sum_{n\in N} c^{lsh}P_{(s,n,t)}^{lsh}]\} \quad (1)$$

subject to:

$$PG_{(g)}^{min}u_{(s,g,t)} \le PG_{(s,g,t)} \le PG_{(g)}^{max}u_{(s,g,t)} \quad \forall\, s\in S, g\in G, t\in T \quad (2)$$

$$P_{(s,n,t)} = \left[\sum_{g\in G_n} PG_{(s,g,t)} + P_{(s,n,t)}^{lsh}\right] - \left[P_{(s,n,t)}^{d}\right] \quad \forall\, s\in S, n\in N, t\in T \quad (3)$$

$$-F_{(m)}^{max} \le F_{(s,m,t)} \le F_{(m)}^{max} \quad \forall\, s\in S, t\in T, m\in M_{(s)} \quad (4)$$

$$F_{(s,m,t)} = (\boldsymbol{PTDF}_{(m)} \times \boldsymbol{P}_{(s,t)}) + \sum_{o\in O_{(s,t)}}\left[\left(PTDF_{(m,frm_{(o)})} - \left(PTDF_{(m,to_{(o)})}\right)FC_{(s,t,o)}\right]\right. \quad \forall\, s\in S, t\in T, m\in M_{(s)} \quad (5)$$

$$(\boldsymbol{PTDF}_{(o)} \times \boldsymbol{P}_{(s,t)}) - FC_{(s,t,o)} + \sum_{o'\in O_{(s,t)}}\left[\left(PTDF_{\left(o,frm_{(o')}\right)} - \left(PTDF_{\left(o,to_{(o')}\right)}\right)FC_{(s,t,o')}\right] = 0 \quad \forall\, s\in S, t\in T, o\in O_{(s,t)} \quad (6)$$

$$\sum_g\left(PG_{(s,g,t)}\right) + \sum_n\left(P_{(s,n,t)}^{lsh} - P_{(s,n,t)}^{d}\right) = 0 \quad \forall\, s\in S, g\in G, t\in T \quad (7)$$

$$u_{(s,g,t)} = u_{(s',g,t)} \quad \forall\, s, s' \in S \quad (8)$$

$$A_{s,t}(x_{s,t}, v_{s,t}, u_{s,t}) \le 0, \quad \forall\, s\in S, t\in T \quad (9)$$

$$B_{s,t}(x_{s,t}, v_{s,t}, u_{s,t}) = 0, \quad \forall\, s\in S, t\in T \quad (10)$$

Expression (1) is the objective function to be minimized over all the defined scenarios in the study period. It includes the fuel cost, no-load cost, start-up and shut-down cost related to each generation unit, and the penalized load shedding cost. Note that load shedding is defined as an expensive generation unit (or negative load) at any bus with a defined load.

Equation (2) applies generation maximum and minimum limits, (3) calculates the nodal net injection power with load shedding modeled as a generator. G_n represents the set of generators connected at bus n. Constraint (4) keeps line flows within the acceptable range for each transmission line. Note that, while $M_{(s)}$ can include all lines, it can be any subset of lines that are selected to be monitored. To be sure that the results stay accurate, the constraints of the removed lines from $M_{(s)}$ must be inactive or redundant. However, the quasi-active constraints may be removed by the proposed method, as will be explained later. Executing a post-processing step, described in Section III-E, the quasi-active constraints are detected and added to the SUC. Note that quasi-active constraint refers to those not binding constraints when they are included in the problem but removing them will affect the results as their corresponding variables violate their limits.

For the set of line outages, noted by O, (5) and (6), calculate the power flow for monitored lines of the set $M_{(s)}$, taking into account the effects of line outages. In (5), the first term on the right-side of the equation calculates line power flow concerning the normal network (without line outages) by using the power transfer distribution factor (PTDF) concept [39], while the second term, which is based on flow-canceling transaction concepts [28], considers the effect of outages of lines in $O_{(s,t)}$ on the power flow of each line in $m \in M_{(s)}$. $FC_{(s,t,o)}$ in (5), is an imaginary power flow variable in flow-canceling transaction method and its calculation depends on the operation point of the system. Note that, at the same time, the solution of the SUC problem depends on the effect of line outages through FCs, while for the calculation of each FC the system operation point should be known. Hence, it is not possible to calculate the $FC_{(s,t,o)}$ outside the SUC problem. To overcome this challenge, a system of equations and unknowns is added to the SUC as a constraint which is presented by (6). In (6), the variables FC depends to each other through arrays of $\boldsymbol{PTDF}$ matrix. In accordance with flow-canceling transaction methodology which assumes a pair of imaginary buses each with its imaginary power injection (FC), constraint (6) adds two equations to the problem regarding each line outage.

Constraint (7) applies power balance in the network. Constraint (8) forces the commitment status to be the same within all the defined scenarios (commitment variables are modeled as first-stage variables). It should be mentioned that (2) to (8) represent the constraints of interest in this study, while other standard constraints, such as ramping up/down, minimum up/down times for generators, constraints regarding allowed values of load shedding and other network constraints are considered as well. Those equality and inequality constraints are modeled through (9) and (10), respectively. Readers are referred to [27] for the complete formulation and algorithm description.

When the abovementioned set of equations is used for large-scale networks with multiple outages, the number of variables and constraints becomes extremely large. Notably, (5) and (6) are the main equations responsible for growing constraints in numbers and complexity, as they combine standard power flow calculations considering the effects of outages. In some studies, such as in [40], the authors suggest combining (5) and (6) into one simpler equation based on detecting parallel lines and lines with power flows within their acceptable limits. However, these techniques consider only one outage and are not applicable for multiple line outages considered in this paper.

A promising fact in solving the SUC problem is that, while the original problem includes many variables and constraints,

not all variables must be calculated within the optimization process, nor all the constraints reach their limits. This suggests that if it is possible to distinguish between the essential variables and constraints to be included in the optimization problem and those that are not, the problem can be solved faster. It is worth mentioning that variables and constraints excluded from the optimization problem can be calculated outside the optimizer to verify the complete set of results (post-processing step).

B. Machine Learning Concept

Machine learning has been widely adopted across various applications, over the recent past, due to the advancements both in computational power and algorithms [41], [42]. It is expected that increasing amounts of data, analyzed by ML methods, can provide solutions of high accuracy and increased speed in power system problems, such as UC [43].

Supervised machine learning algorithms can learn through examples known as observations. Each observation consists of inputs paired with the corresponding output(s) [44]. The training algorithm searches for patterns and correlations between inputs and outputs. After training, a supervised learning algorithm for any new unseen inputs predicts the outputs. In its basic form, a supervised learning algorithm can be formulated as:

$$Y = f(x), \tag{11}$$

where Y represents the predicted output(s) that is determined by a mapping function, f, over the value of x. The mapping function used to connect inputs/features to a predicted output(s) is created during the training process. Ideally, with enough training data to cover the whole operating spectrum of the SUC problem, it is possible to train a perfect supervised mapping function, f, as shown in (11). If such an ideal model existed and could be trained, providing the input data, such as network information, expected uncertainties, and demand data, any desired Y would be acquired rapidly with no need to run SUC again. However, as repeatedly shown in the literature, achieving such an ideal model would be very challenging, if not impossible.

III. Feasibility Study

This section aims to evaluate different possible ML applications to solve or to assist in solving the SUC problem with the formulation as described in (1)-(10). As ML is defined based on inputs and output(s) pairs, this section evaluates different inputs and outputs variables to discover the most suitable candidates for consideration as input features and outputs for the ML model. As a feature engineering process, some candidates are rejected, and the list is narrowed down to a few primary candidates. Next, a feasibility analysis is performed for each candidate to determine the best options.

A. SUC cannot be Replaced by Machine Learning

As the ML model must be trained with a set of solved SUC cases, the accuracy of the trained ML can never be better than the original SUC solution. Hence, the primary motivation for using a trained ML model can be only to reduce the calculation time and/or hardware requirements while maintaining the same accuracy as the original SUC. As a nature of an ML training process, a large number of variables require a large number of solved cases (known as observations) to train the ML model. Since the solution time of SUC is long, obtaining a large enough data set to train the ML becomes a bottleneck. On the other hand, if the SUC problem's solution time is short enough, there is no reason to use ML.

Due to strict SUC constraints, and because it is unlikely to train an ML model to perfectly predict the optimal solution of SUC, a potential alternative is to use machine learning as an assistant to SUC to facilitate the solution process. This way, the result is accurate as of the original SUC, but perfect ML performance is unnecessary. This would translate into a reduced training dataset. Hence, the objective is to use the trained ML to guess/predict the entire or a part of the final solution.

The predicted solution by ML can then be implemented into the SUC as a **warm-start** (also known as advanced start or MIP-start) to solve the case accurately and at increased speed. However, before making it possible to use trained ML to predict the output, one more question should be answered: what inputs can be used as input features, and what outputs should be predicted to help solve the SUC problem, if possible at all? The next subsections offer some insights into this question. While it is not suggested to train a perfect ML model to predict the SUC solution accurately in this paper, training ML to estimate optimal power flow (OPF) to be used as a warm-start without including binary variables is suggested by other studies as well, such as [45], [46].

B. Candidate Inputs and Outputs

In SUC, the desired outputs are the commitment status (binary), the scheduled generated power (continuous) and line flow (continuous). As the power network topology changes slowly over time, it is probably best to train the ML model(s) for a network structure covering the network topology and generators' data. In such a model, data expressing uncertainties are the main input features used to train ML models. For example, assume that 100 lines in the network are vulnerable to failure due to a hurricane. To train the ML model, thousands of SUC problems, known as observations, should be defined and solved, each of which represents a possible hurricane affecting the failure probabilities of those 100 lines. Then, the trained model can predict the SUC solution in response to any unseen hurricane.

In preventive operation, in case of an imminent hurricane, continuous variables are defined as second-stage variables. Since second-stage variables are scenario dependent, there are many more continuous variables compared to binary variables. Thus, considering the nature of ML, continuous variables are harder to predict, requiring more solved cases for training. Moreover, second-stage variables such as generation dispatch can be easily calculated once the first-stage variables are known. Hence, ML can predict the binary commitment variables, while an economic dispatch can be employed to calculate

generated power and line flow quickly with known commitment variables. Note that other binary variables, such as start-up and shut-down, can be derived from the commitment solution.

Alternatively, it is also possible to define another set of fictional binary variables that determine whether each constraint (such as line flow limits) is binding or not. Using these binary variables, it is possible to reduce the number of applied constraints to the problem. As a reduced number of constraints translates into lower calculation time, these binary variables are assumed as candidate outputs for the trained ML model(s). It is worth mentioning that when working with binary variables, the supervised ML classification seems to be a good fit, as binary variables can be translated into two classes, 0 and 1.

C. Test Cases and Software Selection

In this study, a synthetic grid on the footprint of South Carolina with 500 buses, 597 lines, and 90 generation units is used as the test-case. The complete system information can be found in [47], [48]. For the load profile, we use a daily load profile, as in [49].

The main code that handles the MIP-SUC problem is developed on the Java platform through ECLIPSE IDE [50] and uses IBM CPLEX optimization studio ver. 12.10 [51] as the solver. The whole software-package runs on a system with 128 GB of memory and AMD 3900X as the processor unit.

D. Commitment Status as Output

This subsection investigates the computational time savings if generator commitment variables could be predicted. The feasibility test assumes that a trained ML exists that predicts the commitment status at different accuracy levels from perfectly accurate to partly inaccurate. Then, the predicted commitment is implemented as a warm-start to SUC, and the accuracy of the results and solution time are compared with cold-start SUC. In this paper, cold-start SUC refers to the original SUC problem when the solver starts from its default values to solve the problem, and no variable is predicted and applied to the solver. This way, it is possible to evaluate how much time can be saved by predicting commitment status with various error levels.

In general, a larger number of scenarios may result in a better quality of the solution. As determined in [8], however, when line outages are the main source of uncertainty, increasing the number of scenarios from 10 to 17 for a 2,000 bus network does not significantly improve the quality, although it substantially increases the calculation time and hardware requirements. Moreover, the authors found out that the number of scenarios does not change the feasibility study's general conclusion. Hence, as the base case, it is assumed that the original SUC consists of 10 scenarios, ten lines with failure chance. This takes 169 seconds to solve. Next, by using the commitment obtained by solving the original SUC (serves as the perfect prediction), a certain percentage of generator statuses is randomly changed to simulate errors in predictions by the trained ML. Next, different prediction accuracies of commitment status are used to solve the SUC problem as warm-start models**.** While all cases result in accurate optimal value for the objective function, the solution time is highly sensitive to errors in the predicted commitment variables. For each level of accuracy in prediction, ten random cases are solved, and the solution times are reported in Table I. If commitment status could be predicted with 100% accuracy, the solution time would be reduced by 4.7%, which is not significant.

On the other hand, even small errors in status prediction (more than 0.3%) increase the solution time compared to the original SUC. This can be justified considering the time CPLEX needs to implement warm-start, verify if the solution is feasible, and then calculate the other variables of the problem. In case the provided warm-start solution is not accurate, CPLEX will try to fix the solution. The overall time for implementation, verification, and repairing the solution in cases with errors, is longer than required to calculate a first feasible solution with cold-start.

TABLE I
SOLUTION TIME OF SUC CONSIDERING DIFFERENT LEVELS OF ERROR IN THE PREDICTION OF THE COMMITMENT VARIABLES

Case	Solution Time (Sec.)
Original SUC	**169**
Assisted with commitment variable with 0.0% error	***161***
Assisted with commitment variable with 0.1% error	***161~163***
Assisted with commitment variable with 0.2% error	***163~167***
Assisted with commitment variable with 0.3% error	***176~180***
Assisted with commitment variable with 0.5% error	***182~188***
Assisted with commitment variable with 1.0% error	***188~193***
Assisted with commitment variable with 5.0% error	***194~201***
Assisted with commitment variable with 10.0% error	***194~202***
Assisted with commitment variable with > 10% error	***197~205***

E. Suspected Limit Violating Lines as Output

As mentioned before, if the trained ML can predict constraints that are not binding, those constraints can be removed from the model without impacting the final solution. Note that as mentioned in [35], the final solution must be checked for quasi-active constraints removed from the problem. Equations (4) to (6) enforce constraints to the problem that are more complex than others. Hence, a trained ML that removes unnecessary constraints in these three equations can reduce calculation times. In the following, it is assumed that a trained ML model can predict which line violates its thermal limit, represented by M in the formulation. The feasibility study is done in the following steps:

1- SUC is solved with only suspected lines included in (4) to (6). The effect of the accuracy of these suspected lines is discussed later.
2- Using results from step-1, power flows are calculated, and all line flows are compared with their corresponding limits to find if any violation exists. Note that violations also may include quasi-active constraints that may have been removed from the subset of lines.
3- In case of violation, lines with flows exceeding their thermal limits are added to M (constraints (4)-(6) related to these lines are added to the optimization problem).

4- Steps 1 to 3 are done iteratively until there are no lines with flows exceeding their thermal limits.

Like the previous part, to simulate trained ML models with different accuracy levels, random lines are added to or removed from the perfectly predicted lines to be monitored. Table II presents the average times over ten simulations. Note that, original SUC problem is the same as the one used in Table I. In Table II, **FN** is the abbreviation for **F**alse-**N**egative (FN) and means that lines that should have been predicted as suspected to violate their limits have not been predicted. In cases with FN, the SUC model should be solved at least one additional time, as at least one line violates its limitation in the first iteration.

According to Table II, predicting suspected lines with good accuracy can save significant computational time, while the quality of the final solution is the same as the original solution. Moreover, saving is not as sensitive to errors as it was with the commitment variables. Thus, predicting lines suspected to violate their limits seems to be an appropriate candidate output of the trained ML. Table II also suggests that the ML training process should be tuned to prevent FN predictions even at the cost of more **FP** (**F**alse-**P**ositive) errors. As an example, in Table II, 25% of error, including FN, is worse than 50% error without FN. It is worth mentioning that this finding is consistent with industry practices. Industry implementations of unit commitment take a list of monitored lines as their input. If such a list is not available or not comprehensive, the solver uses heuristics to quickly identify the binding constraints [27]. However, the major difference between a regular unit commitment and the SUC is that the set of binding constraints cannot be straightforwardly identified. This is because many transmission lines may fail, and their failure can only be predicted in a probabilistic manner.

TABLE II
SOLUTION TIME OF SUC CONSIDERING DIFFERENT LEVELS OF ERROR IN THE PREDICTION OF SUSPECTED LINES

Case	Solution Time (Sec.)	
	without FN	with FN
Original SUC	*169*	
Assisted, suspected lines with 0.0% error	**8**	
Assisted, suspected lines with 10% error	**25**	**48**
Assisted, suspected lines with 25% error	**51**	**98**
Assisted, suspected lines with 50% error	**87**	**187**
Assisted, suspected lines with 60% error	**104**	**197**
Assisted, suspected lines with 75% error	**134**	**256**
Assisted, suspected lines with 100% error	**308**	

F. Input Features

As explained before, the input features for ML training include data related to uncertainties. In SUC, uncertainties are represented by scenarios. As the prime goal is to assist in solving the SUC problem, the input features should be scenarios or data derived from them.

Note that the part of data that is the same for all scenarios carries no useful information for predictions. For example, if 100 lines out of 597 lines in a network are considered vulnerable to extreme weather, some of these 100 lines will have a non-zero failure probability in different conditions. If any of these conditions are modeled in the SUC problem, the only difference between scenarios should be related to those 100 lines and not all the 597 lines. Hence, scenarios regarding different conditions should be worked out before used as input features.

IV. MACHINE LEARNING TRAINING AND PREDICTION ALGORITHMS

Training ML models: The previous section determined that predicting the set of suspected lines to violate their capacity limit is an appropriate output of the ML model, and scenarios should be used to form input features. To train any ML model, a set of observations should be created. In this study, each observation represents inputs and corresponding outputs related to a possible hurricane that affects the network, for which temporal line failure probabilities can be calculated. Each observation can be defined as a multi-scenarios SUC problem. However, it seems impractical and inefficient to train ML models based on multi-scenarios SUC. This is because the number of engaged variables increases with the number of scenarios, translating into a larger number of required observations.

Moreover, solving each observation takes a long time. Also, if an ML model is trained for a specific number of scenarios, it might not be effective for SUC with any other number of scenarios in its original form. For example, authors estimated that with a network as large as 2,000 buses and 3,200 transmission lines, and assuming less than 10% of lines are vulnerable to failure, the number of required observations for SUC with ten scenarios is more than 100,000 and considering that each SUC takes 20 minutes to be solved (tested on an advanced workstation with 128GB of memory and CPU with 24 Cores), the total required time is **four years**.

Here, an alternative method to train ML models is suggested to reduce the calculation time and make it practical for SUC with any number of scenarios. The idea comes from the fact that when multiple scenarios are created representing a single hurricane, scenarios represent different possible damage levels provoked by that hurricane. The minimum number of scenarios can be one in which that single scenario is the worst-possible case. For a larger number of scenarios, the level of damage modeled within each scenario is between no damage to maximum possible damage. While scenarios represent different levels of damage by a single hurricane, it is possible to assume that each scenario represents maximum damage by a different hurricane. For example, the worst possible scenario of one hurricane can be the second-worst or n^{th}-worst possible scenario of another hurricane of higher intensity. With this assumption, it is possible to train ML models based on single scenario problems, representing the worst-possible case of each observation, and later use the trained models to predict suspected lines for each scenario of the SUC to be solved individually. This is also consistent with the provided formulation, (1)-(10), as, in the formulation, each scenario has its own set of suspected lines that violate their capacities.

To create a set of observations, the following steps are implemented:

1. For each hurricane (observation), create the worst-case scenario, i.e., the one with the highest possible number of failures. The solution of a deterministic UC for this worst-case scenario will identify the lines that reach their capacity.
2. Determining Input: For the selected scenario in step 1, any failed element at any time is marked. The input corresponding to the observation is a vector with entries set as 1 if the element is marked, and 0 if not. The size of the input vector is equal to the number of vulnerable elements.
3. Determining the Corresponding Output: Solve the UC problem with the worst-case scenario from step 1, and mark those lines that reach their capacity at any time of the examined horizon. The output corresponding to the observation is a vector with a size equal to the total number of lines. In this vector, any line marked as congested will be assigned a value of 1, and 0 if not marked. While this procedure removes the dependency of output on time (i.e., load profile) as a capacity violation of any line at any time triggers the corresponding variable, it will automatically consider the worst possible condition (in term of lines reaching their capacity) and reduces the chance of FN error.
4. Repeat steps 1 to 3 for all observations. Each observation includes input and its corresponding output vectors.
5. Size-Reduction: As mentioned before, if any of the variables does not change over different observations, it does not carry any useful information in ML training. Hence, if a line never reaches its capacity over all the observations, it should be removed from the outputs. Similarly, if an element never fails, it should be removed from the input set.

After the abovementioned steps, the input matrix with rows as observations and columns as features, and output matrix with rows as observations and each column as output are created.

It should be mentioned that the ML model used falls in the category of supervised classification. It receives multiple inputs and predicts only one output for each observation. Hence, it is necessary to train one ML model for each output (line included in the outputs after applying size-reduction described in step 5), which means the number of trained models is equal to the number of columns in the reduced form of the output matrix.

Prediction Through the trained ML models: To predict the desired outputs for an unseen event/hurricane, the SUC problem should be formed with its scenario set. Next, for each scenario, the input vector should be created, as explained in step 2. Then, this vector of features should be deployed into the trained ML models. The predicted output of each model is either 1 (line should be monitored) or 0 (line does not need to be monitored). After repeating the same process for all the scenarios, $M_{(s)}$ is predicted. Finally, the SUC problem is solved by following the steps presented in Section III-E.

V. Simulation Studies

To demonstrate the efficiency of the proposed ML-assisted SUC algorithm, the same network as in the feasibility study is used in this section to simulate 24 hours. In this paper, 40 out of 597 transmission lines are considered vulnerable to failure due to extreme weather events. While a larger number of lines was included in the vulnerable set, this would divide the network into islands.

Training ML Models: The ML models should be trained with as many as possible different combinations of the status of those 40 vulnerable lines as observations. Equation (12) provides the number of possibilities to select X items out of Y possible options when order does not matter.

$$N = \frac{Y!}{X!\,(Y-X)!} \tag{12}$$

Even without considering when each failure happens, there are more than 10E+12 different combinations of outages. Obviously, it is not possible to solve all these possible combinations to train the ML models. Hence, a limited number of observations should be selected efficiently. To increase the efficiency of the training set, the number of observations regarding the number of outages should follow the normal distribution, as in (12) (as shown in Fig. 1). For example, observations with 20 outages should have the highest number.

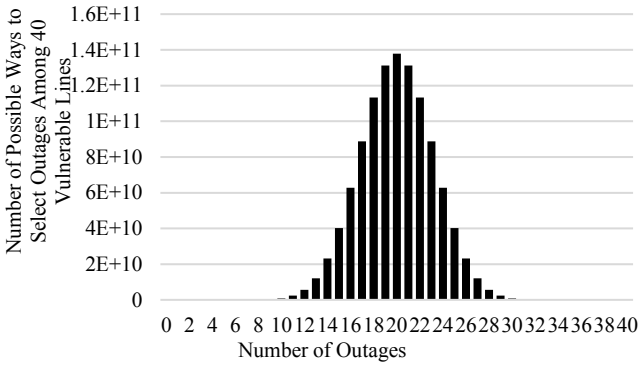

Fig. 1. Graphical representation of (12) when Y equals 40 and X changes from 0 to 40.

To enhance the efficiency further, some exceptions are made, i.e., forty of the selected observations are reserved for an outage of a single line, one observation is reserved for the case with no outage, and one observation for the case of all 40 outages. After the number of observations for each value of X is determined, the outages and their time of failure are generated randomly based on their failure probabilities. Next, a single scenario SUC is solved to create the output corresponding to each observation.

An adequate number of observations is identified as the minimum number of observations, for which the accuracy of the trained model reaches saturation. For the test-case, the accuracy of the trained ML model reaches a satisfactory performance with 5,000 observations. However, to ensure high accuracy, 20,000 observations are used.

The problem is modeled as a supervised classification ML model, defined as "Ensemble Method: Bagged Trees," with the number of learners equal to 100 and the maximum number of splits of 8,999. 10% of the observations are used for validation to prevent over-fitting, and 5% of observations are kept to assess the accuracy of the model after training. The total accuracy

of the trained model when it was used for 1,000 unseen cases is 99.3%. Detailed accuracy results are shown in Fig. 2. As is shown, only 3 FN (0.3%) and 4 FP (0.4%) predictions were observed. It is worth mentioning that 2 out of 3 FN predictions were related to the same line, line "183", which is a line that violates its capacity only in 5 out of 20,000 observations used for ML training; thus, it has a high chance of FN prediction.
It is worth mentioning that after the dataset is created (in 5 hours of computational time), the overall training time is less than 180 seconds, and most of it is spent on loading the large dataset of observations from the hard drive into the trainer.

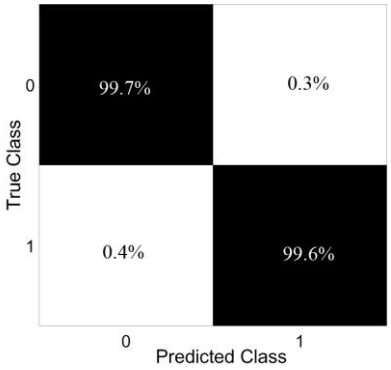

Fig. 2. Confusion matrix for the trained ML model. Black represents correct prediction and white incorrect prediction.

Using the Trained Model to Assist Unseen SUC Problem: Once the training process is complete, the trained models are ready to assist in solving any unseen SUC problem. To evaluate the effectiveness of the proposed method, different SUC problems are created and solved with both methods: Original SUC and ML-Assisted SUC.

The SUC problems are created with different numbers of line outages (i.e., different hurricane intensities). They cover zero to 30 possible outages, where each outage may happen at any random time of the examined horizon. The scenario set regarding each SUC problem consists of ten scenarios. The optimization problem is formulated as described briefly in Section II and explained in detail in [27]. Fig. 3 illustrates the solution time for both methods and different SUC problems (for every possible number of line-outages, ten randomly generated problems are solved). As shown, the solution time with the ML-Assisted method is significantly reduced compared to the original SUC solution.

More specially, the ML-Assisted model reduces the calculation time by 90% on average (within 75% minimum and 97% maximum time reductions) for the different SUC problems. Simultaneously, the value of the objective function, commitment status, and line flows are similar in both methods within a defined MIP gap of 1% in CPLEX.

Another measure of the performance of the proposed algorithm is the accuracy of trained ML in predicting suspected lines to violate their capacity. Among different SUCs, the trained ML model perfectly predicted the suspected lines for all scenarios. While no FN was observed within the simulations in this section, even for cases with FN, when post-processing reveals a line violates its capacity, the total calculation time is still much less than the original SUC. Moreover, to show the effect of predicting suspected lines on the complexity of SUC, the number of implemented constraints is shown in Table III for the original and ML-Assisted models. This clearly shows the reason for the significant reduction in computation time with the proposed method.

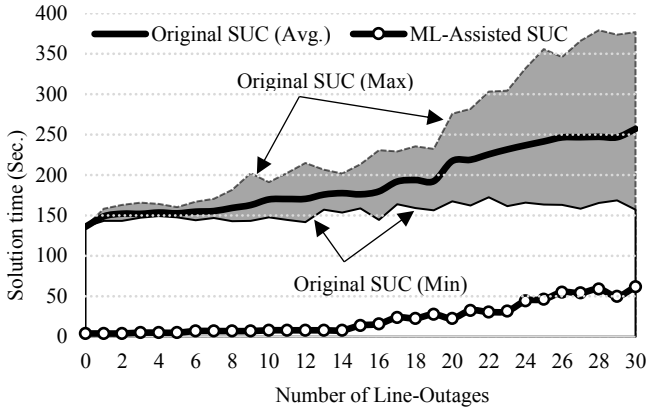

Fig. 3. Solution time for the original SUC and the proposed ML-Assisted SUC

TABLE III
COMPARISON OF ORIGINAL SUC AND ML-ASSISTED SUC: NUMBER OF IMPLEMENTED CONSTRAINTS CORRESPONDING TO TRANSMISSION LINES

Number of Line Outages	Number of Constraints Regarding Transmission Lines	
	Original SUC	ML-Assisted SUC
0	143,280	720
5	143,574	1,014
10	144,084	1,524
20	144,608	2,888
30	145,416	3,936

Effect of Load Profile: The ML model is trained based on the default/predicted load profile. To investigate the impacts of different load profiles on the efficiency of trained models, a new load profile is used, with a maximum demand all the time. Similar to Fig. 3, solution times for both methods and all test cases are shown in Fig. 4. The ML-Assisted SUC is superior in all cases. Moreover, even in this extreme case, the accuracy of predicting suspected lines remains the same. Thus, the proposed method can be trained with load earlier forecasts and used with near-landfall forecasts. This way, the changes in the load profile will be minimal. Note that fluctuations in Fig. 4 are results of randomly selected times that lines fail.

Effect of Congestion: The effect of different congestion levels is evaluated by derating the capacity of all lines to 90% of their original capacity. Using the reduced transmission capacities, the ML training process is repeated. With the original operating limits, 63 lines reached their capacity during the entire training process. From 63 observed lines, 31 reach their capacities in more than 1% of observations and 17 in less than 0.1% of the observations. When the line capacities were reduced to 90%, the total number of suspected lines increased to 88 lines, with 42 of them in more than 1% and 16 lines in less than 0.1% of observations. Fig. 5 represents a histogram of the first 25 highly observed lines in both normal and congested networks.

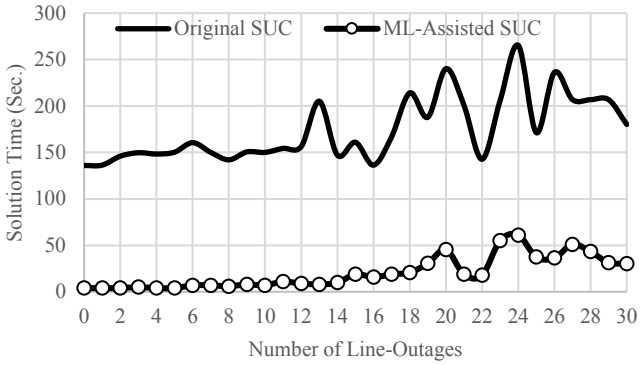

Fig. 4. Solution time for the original SUC and the ML-Assisted SUC when demand is at its maximum all times.

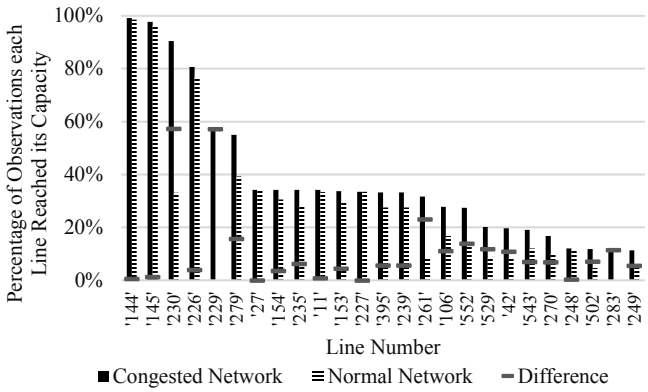

Fig. 5. Histogram of the first 25 highly observed lines reaching their capacities for normal operation and increased congestion cases.

As expected, the number of observations of lines that reached their capacity increases in the congested network. There are two lines, '229' and '283', that never reach their capacities in normal operation, contrary to the congested network. Finally, the average accuracies of the trained models are almost the same, with 99.3% accuracy for the original network and 99.0% for the congested network. This indicates that the proposed method keeps its performance when network congestions are increased.

A possible solution to reduce the chance of FN error is to train the MLs with lower line capacity limits and use them for the original network. This is evaluated by using two sets of trained MLs, one trained with normal and the other with lower capacity limits. Both are used to solve the same 1,000 unseen problems with normal network conditions (no increase in congestion). While the overall accuracy stays almost the same (99.2% with the congested network in comparison with 99.3% accuracy of the original model), the FN error rate reduced to 0.2%, and FP error rate increased to 0.6%. As the effect of FN on solution time is much higher than FP, this practice can be recommended, especially if the accuracy of the trained ML models is relatively low.

Final Remarks: The rest of this section clarifies a few potential questions and remarks.

Remark 1: Since the ML model is trained based on single scenario UCs, why not solving that single scenario UC before the original SUC to determine the violations instead of predicting them through ML? To address this question, the authors have tested the same cases with both methods: one with ML-assisted SUC and the other with detecting the violations by solving a single scenario UCs before solving the main SUC. Fig. 6 demonstrates that ML-assisted SUC is always faster. The difference between the methods increases as the number of outages increases. The reason for the faster solution of the ML-assisted SUC is that although the ML model is trained based on a single scenario UC, it can predict the lines for all individual scenarios of SUC. We would like to clarify that the same set of suspected lines is not used for all the scenarios within the scenario set; rather, each scenario independently obtains its own set of estimated binding constraints using ML.

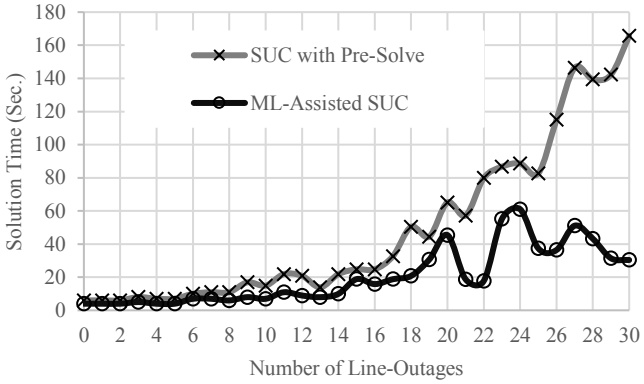

Fig. 6. Comparison between ML-assisted SUC and SUC with single, worst-case UC solved before the main SUC.

On the other side, pre-solving single scenario UCs to determine the suspected lines regarding all the scenarios in the set requires solving the single scenario UCs as many times as the number of scenarios in the SUC problem. While it is possible to solve the UC for only the scenario with the highest number of outages within the scenario set, this will apply an unnecessarily large number of suspected lines to the rest of the scenarios and increase the overall solution time of SUC. While pre-solving the single UCs can reduce the total solution time in comparison with the original SUC problem, it is always inferior to the ML-assisted SUC, with its near-instantaneous predictions.

Remark 2: An alternative method to the proposed ML-assisted model may be to solve the SUC problem by considering only the worst-case scenario and then use the calculated commitment variable as the first stage variable to serve as warm-start values to solve the original SUC problem. The motivation is that while solving a single scenario UC needs its own additional time, solving SUC with a feasible solution as warm-start can save more time to compensate that additional time and reduces the total calculation time.

The authors used the same test cases as in the previous sections to evaluate this idea. It was shown that using commitment variables obtained from the worst-case as a warm-start for the SUC increases the total calculation time significantly. As an example of the calculations, solving the original SUC problem with 20 outages needs 205 seconds. The UC, with only the worst-case scenario, needs 26 seconds to be solved. After that, the warm-start SUC needs another 276 seconds to be solved. The total calculation time is increased significantly from 205

seconds to 302 seconds. The same pattern was observed with any other case with outages. The calculation time with warm-start is increased in comparison with the original SUC because the commitment corresponding to worst-case is too conservative for other scenarios. As a result, the objective value is far from its optimal value. CPLEX tries to fix the provided solution. However, fixing a solution far from the optimal one takes more time than cold start.

Remark 3: In the dataset production, (12) was used to determine the maximum number of outages for each observation, while the rest of the dataset was generated randomly for time, location, and the possibility of each outage. A possible question is if any improvement of the proposed method can be achieved by a totally random generated dataset of observations.

To assess this, two sets of observations, one based on the proposed algorithm presented and one randomly generated. The same supervised classification is used to train both ML models. A comparison of the results shows that the same set of suspected lines is determined. However, for some of the lines, the number of observations in which they reached their capacity is slightly different.

To compare the two trained MLs, two factors are compared; the average accuracy of prediction and accuracy of prediction related to important lines. Important lines are lines that will most probably reach their capacity in the highest number of unseen cases. These are shown in descending order in the histogram of Fig. 5. The average accuracy of MLs for all the suspected lines with the proposed method is 99.3%, and the accuracy with random observations is 97.2%. In terms of individual accuracies related to each line, Fig. 7 represents the results for 20 of the most observed lines to reach their capacity (left side of the histogram). The proposed method enhances the total average accuracy of the trained model in comparison with randomly generated observations.

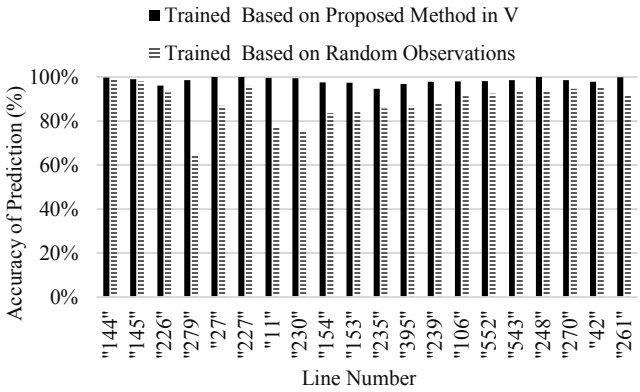

Fig. 7. Accuracy of trained MLs when predicting each individual line

Remark 4: There are not many papers in the literature addressing the problem of this paper. In order to prove the superiority of the proposed method, results are compared with the most relevant method proposed in [35]. In [35], the authors suggest the use of historically recorded information to analyze the operation of the network and exclude inactive and redundant constraints corresponding to thermal violations from the SUC formulation. The historical data includes the system state (e.g., power flows) under several scenarios of known net demand. They determine (predict) lines with active constraints in the UC problem by the *K*-nearest neighbors (*K*NN) technique to find the closest historical system states that best represent the current system state in terms of load demand. The distance between the historical system states and the current state is calculated based on the net demand of each bus, using the **PTDF** as weighting factors.

The main difference with the method proposed in this paper is that the method in [35] does not consider any change in the network topology (line outages are not considered) over the study period, and the *K* neighbors are determined in terms of nodal demand. In the proposed method in the current paper, the network topology changes dramatically, and the *K*-nearest neighbors should be determined by the lines' status (online or damaged). Another assumption in [35] is that data about the status of the network are available, which for the application of the current paper, not enough data is normally available. Therefore, in order to make a fair comparison between the two methods, it is assumed that data are available for the different network topologies. These are generated through simulations of different possible hurricanes and are the same as the ones used in this study to train the ML model.

To implement the method of [35], the following steps are taken:

1. A vector $\widehat{\boldsymbol{d}}$ with a dimension of L is defined for each historical system state. In general, L is the number of lines, and in compact form, it can be the number of at-risk lines to a hurricane. Each array of $\widehat{\boldsymbol{d}}$, $\hat{d}_l$, is 1 if line l is online and 0 otherwise. For each historical system state, the non-congested lines are also recorded. Note that, historical system state for this method is the same as observation for the ML model.
2. For each simulated historical state of the system, a unit commitment problem is solved, and statuses of all lines (congested or not) are determined. Line statuses vector, $\boldsymbol{S}$, is the same as in [35].
3. Pairs of ($\widehat{\boldsymbol{d}}, \boldsymbol{S}$) are used in the *KNN* method to determine the closest historical state when $\boldsymbol{S}$ should be predicted for an unseen case (hurricane), $\boldsymbol{d}$.
4. The *K*-nearest neighbors of each scenario are determined based on the Euclidean distance between $\boldsymbol{d}$ and $\widehat{\boldsymbol{d}}$. As the outages of lines are the main uncertainty variable instead of net demand, the Line Outage Distribution Factor (LODF) is used as a weight factor (instead of PTDF).
5. Note that, as the outage of lines and also the congestion status of lines are modeled as binary variables, the dependency of the model to time is removed from all equations in [35].

Fig. 8 is used to compare the accuracy of both methods when predicting suspected lines to reach their capacities. Ten different unseen hurricanes are applied to the network, and the original SUCs are solved to determine the true lines that reach their capacities. Then the same hurricanes are used in both methods, and each method predicts its own set of suspected lines. Results

are compared in terms of total average accuracy, FN and FP error rates. In Fig. 8, lines on the horizontal axis are sorted based on their probabilities to reach their capacities (going from left to right, the chance of violating capacity goes lower by each line). For example, line '144' was observed to reach its capacity in almost all the observations (within 20,000 randomly generated), while line '183' only reaches its capacity in 4 observations (rate of 0.02%).

The proposed method in this paper predicted 37 out of 38 occasions correctly, with 1 FP (line '395', hurricane '9'), and 1 FN (line '183', hurricane '1'). The modified version of the method in [35] predicted 26 out of 38 correctly and had 1 FP and 12 FN. As seen, the rate of FP is the same, however, the FN error which significantly affects the solution time of the SUC is much higher with [35]. This is the main advantage of the proposed method for the under-study problem with severe weather conditions. It should be mentioned that, as recommended in [35], larger values for K parameter in KNN reduce the rate of FN errors, however, this comes at the cost of a higher FP rate. In any case, for K=[5-100], the method in [35] never reached the accuracy of the proposed method in the current study. Overall, within 1,000 unseen hurricanes, the total accuracy of prediction by [35] was 95.9%, with 0.4% of FP rate and 3.7% of FN rate.

It should be noted that for a high number of outages, the proposed method will be much faster than the method in [35] because of the much lower FN rate. Note that normally, as the number of outages increases, the chance of violating thermal capacity for lines increases, which means an increase in FN rate. As an example, the average calculation times for SUC with 5 outages are 5 seconds with the proposed method and 5.5 seconds with the method of [35], while for 30 outages, the corresponding times are 62 seconds and 132 seconds, respectively.

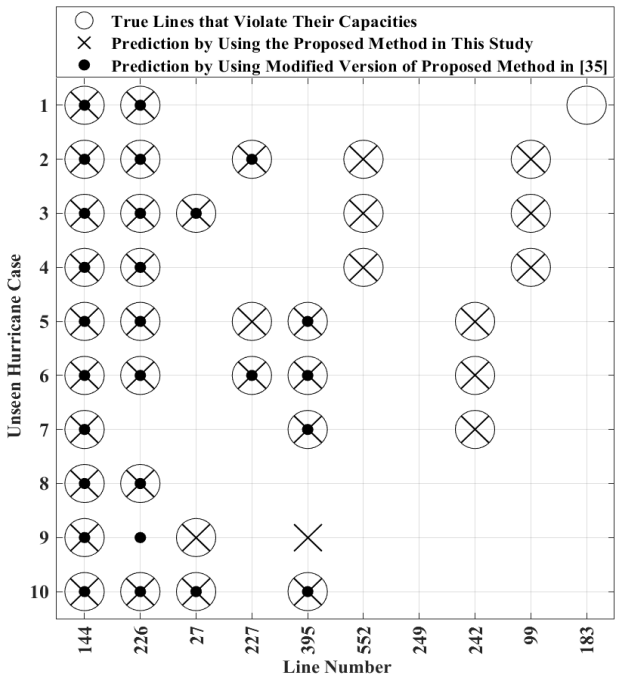

Fig. 8. Comparison of accuracy of the proposed method in this study with a modified version of the proposed method in [35]

VI. Conclusion

For solving the unit commitment problem in modern power systems, multiple sources of uncertainties must be considered. Stochastic unit commitment can effectively tackle uncertainties and offer an efficient solution, but at a high computational cost. This computational burden limits the applicability of stochastic unit commitment. This paper develops a novel method based on machine learning to reduce the computational time associated with preventive stochastic unit commitment. In particular, the trained model predicts the constraints that can be removed from the original problem, thus speeding up the solver. The proposed method offers the same accuracy as the original stochastic method while significantly reducing the calculation time. Simulation studies show that the reduction in calculation time is 93% on average, compared to the original stochastic method, demonstrating the effectiveness of the proposed method.

References

[1] B. Saravanan, S. Das, S. Sikri, and D. P. Kothari, "A solution to the unit commitment problem-a review," *Front. Energy*, vol. 7, no. 2, pp. 223–236, 2013, doi: 10.1007/s11708-013-0240-3.

[2] A. Bhardwaj, V. K. Kamboj, V. K. Shukla, B. Singh, and P. Khurana, "Unit commitment in electrical power system - A literature review," in *2012 IEEE International Power Engineering and Optimization Conference, PEOCO 2012 - Conference Proceedings*, 2012, pp. 275–280, doi: 10.1109/PEOCO.2012.6230874.

[3] B. Stott, J. Jardim, and O. Alsaç, "DC power flow revisited," *IEEE Trans. Power Syst.*, vol. 24, no. 3, pp. 1290–1300, 2009, doi: 10.1109/TPWRS.2009.2021235.

[4] S. Takriti, J. R. J. R. J. R. J. R. Birge, and E. Long, "A stochastic model for the unit commitment problem," *IEEE Trans. Power Syst.*, vol. 11, no. 3, pp. 1497–1508, 1996, doi: 10.1109/59.535691.

[5] P. Carpentier, G. Gohen, J.-C. Culioli, and A. Renaud, "Stochastic optimization of unit commitment: a new decomposition framework," *IEEE Trans. Power Syst.*, vol. 11, no. 2, pp. 1067–1073, May 1996, doi: 10.1109/59.496196.

[6] Q. P. Zheng, J. Wang, and A. L. Liu, "Stochastic Optimization for Unit Commitment—A Review," *IEEE Trans. Power Syst.*, vol. 30, no. 4, pp. 1913–1924, Jul. 2015, doi: 10.1109/TPWRS.2014.2355204.

[7] E. Du, N. Zhang, C. Kang, and Q. Xia, "Scenario Map Based Stochastic Unit Commitment," *IEEE Trans. Power Syst.*, vol. 33, no. 5, pp. 4694–4705, Sep. 2018, doi: 10.1109/TPWRS.2018.2799954.

[8] F. Mohammadi and M. Sahraei-Ardakani, "Multidimensional Scenario Selection for Power Systems With Stochastic Failures," *IEEE Trans. Power Syst.*, vol. 35, no. 6, pp. 4528–4538, Nov. 2020, doi: 10.1109/TPWRS.2020.2990877.

[9] H. Quan, D. Srinivasan, and A. Khosravi, "Incorporating Wind Power Forecast Uncertainties into Stochastic Unit Commitment Using Neural Network-Based Prediction Intervals," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 26, no. 9, pp. 2123–2135, Sep. 2015, doi: 10.1109/TNNLS.2014.2376696.

[10] P. Pinson, H. Madsen, H. A. Nielsen, G. Papaefthymiou, and B. Klöckl, "From probabilistic forecasts to statistical scenarios of short-term wind power production," *Wind Energy*, vol. 12, no. 1, pp. 51–62, Jan. 2009, doi: 10.1002/we.284.

[11] Z. Wu, P. Zeng, X. P. Zhang, and Q. Zhou, "A Solution to the Chance-Constrained Two-Stage Stochastic Program for Unit Commitment with Wind Energy Integration," *IEEE Trans. Power Syst.*, vol. 31, no. 6, pp. 4185–4196, Nov. 2016, doi: 10.1109/TPWRS.2015.2513395.

[12] S. Cordova, H. Rudnick, A. Lorca, and V. Martinez, "An Efficient Forecasting-Optimization Scheme for the Intraday Unit Commitment Process under Significant Wind and Solar Power," *IEEE Trans. Sustain. Energy*, vol. 9, no. 4, pp. 1899–1909, Oct. 2018, doi: 10.1109/TSTE.2018.2818979.

[13] Q. Wang, J. Wang, and Y. Guan, "Stochastic unit commitment with uncertain demand response," *IEEE Trans. Power Syst.*, vol. 28, no. 1, pp. 562–563, 2013, doi: 10.1109/TPWRS.2012.2202201.

[14] B. Analui and A. Scaglione, "A Dynamic Multistage Stochastic Unit Commitment Formulation for Intraday Markets," *IEEE Trans. Power Syst.*, vol. 33, no. 4, pp. 3653–3663, Jul. 2018, doi: 10.1109/TPWRS.2017.2768384.

[15] A. Papavasiliou, S. S. Oren, and B. Rountree, "Applying High Performance Computing to Transmission-Constrained Stochastic Unit Commitment for Renewable Energy Integration," *IEEE Trans. Power Syst.*, vol. 30, no. 3, pp. 1109–1120, May 2015, doi: 10.1109/TPWRS.2014.2341354.

[16] S. Bahrami and V. W. S. Wong, "Security-Constrained Unit Commitment for AC-DC Grids with Generation and Load Uncertainty," *IEEE Trans. Power Syst.*, vol. 33, no. 3, pp. 2717–2732, May 2018, doi: 10.1109/TPWRS.2017.2749303.

[17] Y. Fu and M. Shahidehpour, "Fast SCUC for large-scale power systems," *IEEE Trans. Power Syst.*, vol. 22, no. 4, pp. 2144–2151, Nov. 2007, doi: 10.1109/TPWRS.2007.907444.

[18] P. A. Ruiz, C. R. Philbrick, E. Zak, K. W. Cheung, and P. W. Sauer, "Uncertainty Management in the Unit Commitment Problem," *IEEE Trans. Power Syst.*, vol. 24, no. 2, pp. 642–651, May 2009, doi: 10.1109/TPWRS.2008.2012180.

[19] A. Arab, A. Khodaei, S. K. Khator, K. Ding, V. A. Emesih, and Z. Han, "Stochastic pre-hurricane restoration planning for electric power systems infrastructure," *IEEE Trans. Smart Grid*, vol. 6, no. 2, pp. 1046–1054, Mar. 2015, doi: 10.1109/TSG.2015.2388736.

[20] D. N. Trakas and N. D. Hatziargyriou, "Resilience Constrained Day-Ahead Unit Commitment Under Extreme Weather Events," *IEEE Trans. Power Syst.*, vol. 35, no. 2, pp. 1242–1253, Mar. 2020, doi: 10.1109/TPWRS.2019.2945107.

[21] Y. Sang, J. Xue, M. Sahraei-Ardakani, and G. Ou, "An Integrated Preventive Operation Framework for Power Systems During Hurricanes," *IEEE Syst. J.*, pp. 1–11, Nov. 2019, doi: 10.1109/JSYST.2019.2947672.

[22] A. Nikoobakht, J. Aghaei, and M. Mardaneh, "Retraction: Optimal transmission switching in the stochastic linearised SCUC for uncertainty management of the wind power generation and equipment failures [Gener. Transm. Distrib., 12, 1, (2018) (3780-3792)] doi: 10.1049/iet-gtd.2017.0617," *IET Generation, Transmission and Distribution*, vol. 12, no. 17. Institution of Engineering and Technology, p. 4060, 30-Sep-2018, doi: 10.1049/iet-gtd.2018.0329.

[23] J. Xue, F. Mohammadi, X. Li, M. Sahraei-Ardakani, G. Ou, and Z. Pu, "Impact of transmission tower-line interaction to the bulk power system during hurricane," *Reliab. Eng. Syst. Saf.*, vol. 203, p. 107079, Nov. 2020, doi: 10.1016/j.ress.2020.107079.

[24] S. W. Park, Q. Xu, and B. F. Hobbs, "Comparing scenario reduction methods for stochastic transmission planning," *IET Gener. Transm. Distrib.*, vol. 13, no. 7, pp. 1005–1013, Apr. 2019, doi: 10.1049/iet-gtd.2018.6362.

[25] Y. Wang, Y. Liu, and D. S. Kirschen, "Scenario Reduction With Submodular Optimization," *IEEE Trans. Power Syst.*, vol. 32, no. 3, pp. 2479–2480, May 2017, doi: 10.1109/TPWRS.2016.2603448.

[26] J. Hu and H. Li, "A New Clustering Approach for Scenario Reduction in Multi-Stochastic Variable Programming," *IEEE Trans. Power Syst.*, vol. 34, no. 5, pp. 3813–3825, Sep. 2019, doi: 10.1109/TPWRS.2019.2901545.

[27] F. Mohammadi and M. Sahraei-Ardakani, "Tractable Stochastic Unit Commitment for Large Systems During Predictable Hazards," *IEEE Access*, vol. 8, pp. 115078–115088, 2020, doi: 10.1109/ACCESS.2020.3004391.

[28] P. A. Ruiz, E. Goldis, A. M. Rudkevich, M. C. Caramanis, C. R. Philbrick, and J. M. Foster, "Security-Constrained Transmission Topology Control MILP Formulation Using Sensitivity Factors," *IEEE Trans. Power Syst.*, vol. 32, no. 2, pp. 1597–1605, 2017, doi: 10.1109/TPWRS.2016.2577689.

[29] K. A. Papadogiannis and N. D. Hatziargyriou, "Optimal Allocation of Primary Reserve Services in Energy Markets," *IEEE Trans. Power Syst.*, vol. 19, no. 1, pp. 652–659, Feb. 2004, doi: 10.1109/TPWRS.2003.820702.

[30] E. S. Karapidakis and N. D. Hatziargyriou, "Online preventive dynamic security of isolated power systems using decision trees," *IEEE Trans. Power Syst.*, vol. 17, no. 2, pp. 297–304, May 2002, doi: 10.1109/TPWRS.2002.1007896.

[31] E. M. Voumvoulakis and N. D. Hatziargyriou, "Decision trees-aided self-organized maps for corrective dynamic security," *IEEE Trans. Power Syst.*, vol. 23, no. 2, pp. 622–630, May 2008, doi: 10.1109/TPWRS.2008.920194.

[32] L. Duchesne, E. Karangelos, and L. Wehenkel, "Recent Developments in Machine Learning for Energy Systems Reliability Management," *Proc. IEEE*, pp. 1–21, May 2020, doi: 10.1109/jproc.2020.2988715.

[33] G. Dalal, E. Gilboa, S. Mannor, and L. Wehenkel, "Unit commitment using nearest neighbor as a short-term proxy," in *20th Power Systems Computation Conference, PSCC 2018*, 2018, doi: 10.23919/PSCC.2018.8442516.

[34] G. Dalal, E. Gilboa, S. Mannor, and L. Wehenkel, "Chance-Constrained Outage Scheduling Using a Machine Learning Proxy," *IEEE Trans. Power Syst.*, vol. 34, no. 4, pp. 2528–2540, Jul. 2019, doi: 10.1109/TPWRS.2018.2889237.

[35] S. Pineda, J. M. Morales, and A. Jimenez-Cordero, "Data-Driven Screening of Network Constraints for Unit Commitment," *IEEE Trans. Power Syst.*, pp. 1–1, Mar. 2020, doi: 10.1109/tpwrs.2020.2980212.

[36] G. Dalal and S. Mannor, "Reinforcement learning for the unit commitment problem," in *2015 IEEE Eindhoven PowerTech, PowerTech 2015*, 2015, doi: 10.1109/PTC.2015.7232646.

[37] B. Wang, M. Zhou, B. Xin, X. Zhao, and J. Watada, "Analysis of operation cost and wind curtailment using multi-objective unit commitment with battery energy storage," *Energy*, vol. 178, pp. 101–114, Jul. 2019, doi: 10.1016/j.energy.2019.04.108.

[38] M. Zhou, B. Wang, T. Li, and J. Watada, "A data-driven approach for multi-objective unit commitment under hybrid uncertainties," *Energy*, vol. 164, pp. 722–733, Dec. 2018, doi: 10.1016/j.energy.2018.09.008.

[39] H. Ronellenfitsch, M. Timme, and D. Witthaut, "A Dual Method for Computing Power Transfer Distribution Factors," *IEEE Trans. Power Syst.*, vol. 32, no. 2, pp. 1007–1015, Mar. 2017, doi: 10.1109/TPWRS.2016.2589464.

[40] L. A. Roald and D. K. Molzahn, "Implied Constraint Satisfaction in Power System optimization: The Impacts of Load Variations," in *2019 57th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2019*, 2019, pp. 308–315, doi: 10.1109/ALLERTON.2019.8919809.

[41] S. Marsland, "Chapter 1: Introduction," in *Machine Learning: An Algorithmic Perspective*, Second edi., Boca Raton, FL, USA: CRC Press, Taylor and Francis Group, 2015, pp. 1–11.

[42] Y. Baştanlar and M. Özuysal, "Introduction to machine learning," *Methods Mol. Biol.*, vol. 1107, pp. 1-36o, 2014, doi: 10.1007/978-1-62703-748-8_7.

[43] N. Hatziargyriou, "Machine learning applications to power systems," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 2049 LNAI, pp. 308–317, 2001, doi: 10.1007/3-540-44673-7_20.

[44] Y. Zhang, "Types of Machine Learning Algorithms," in *New Advances in Machine Learning*, Rijeka, Croatia: InTech, 2010, pp. 19–20.

[45] K. Baker, "Learning Warm-Start Points for AC Optimal Power Flow," in *IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2019, pp. 1–6, doi: 10.1109/MLSP.2019.8918690.

[46] F. Diehl, "Warm-Starting AC Optimal Power Flow with Graph Neural Networks," in *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, 2019, pp. 1–6.

[47] A. B. Birchfield, T. Xu, K. M. Gegner, K. S. Shetye, and T. J. Overbye, "Grid Structural Characteristics as Validation Criteria for Synthetic Networks," *IEEE Trans. Power Syst.*, vol. 32, no. 4, pp. 3258–3265, Jul. 2017, doi: 10.1109/TPWRS.2016.2616385.

[48] Adam Birchfield, "ACTIVSg2000: 2000-bus synthetic grid on footprint of Texas," 2019. [Online]. Available: https://electricgrids.engr.tamu.edu/electric-grid-test-cases/activsg2000/. [Accessed: 26-Feb-2019].

[49] F. Mohammadi, M. Sahraei-Ardakani, Y. M. Al-Abdullah, and G. T. Heydt, "Coordinated Scheduling of Power Generation and Water Desalination Units," *IEEE Trans. Power Syst.*, vol. 34, no. 5, pp. 3657–3666, Sep. 2019, doi: 10.1109/TPWRS.2019.2901807.

[50] ECLIPSE Foundation, "Eclipse IDE for Java EE Developers | Eclipse Packages," 2020. [Online]. Available: https://www.eclipse.org/downloads/packages/release/kepler/sr2/eclipse-ide-java-ee-developers. [Accessed: 20-Feb-2019].

[51] IBM, "CPLEX Optimizer | IBM," 2018. [Online]. Available: https://www.ibm.com/analytics/cplex-optimizer. [Accessed: 20-Feb-2019].