Accelerating L-shaped Two-stage Stochastic SCUC with Learning Integrated Benders Decomposition

Fouad Hasan, Student Member, IEEE, Amin Kargarian, Senior Member, IEEE

Abstract—Benders decomposition is widely used to solve large mixed-integer problems. This paper takes advantage of machine learning and proposes enhanced variants of Benders decomposition for solving two-stage stochastic security-constrained unit commitment (SCUC). The problem is decomposed into a master problem and subproblems corresponding to a load scenario. The goal is to reduce the computational costs and memory usage of Benders decomposition by creating tighter cuts and reducing the size of the master problem. Three approaches are proposed, namely regression Benders, classification Benders, and regression-classification Benders. A regressor reads load profile scenarios and predicts subproblem objective function proxy variables to form tighter cuts for the master problem. A criterion is defined to measure the level of usefulness of cuts with respect to their contribution to lower bound improvement. Useful cuts that contain the necessary information to form the feasible region are identified with and without a classification learner. Useful cuts are iteratively added to the master problem, and non-useful cuts are discarded to reduce the computational burden of each Benders iteration. Simulation studies on multiple test systems show the effectiveness of the proposed learning-aided Benders decomposition for solving two-stage SCUC as compared to conventional multi-cut Benders decomposition.

Index Terms—Stochastic unit commitment, Benders decomposition, useful cuts, machine learning.

I. Introduction

A. Background and Motivation

S ECURITY-constrained unit commitment (SCUC) is a complex optimization problem solved daily to determine the optimal schedule of generating units [1]. SCUC is a mixedinteger program (MIP) that involves a mix of continuous and integer decision variables. The challenge with mixed integer problems is that the inclusion of integer variables often makes the problem NP-hard. SCUC becomes even more complex under input parameter (e.g., demand) uncertainty [2]-[4] To counterbalance uncertainty in power system operation, SCUC is often formulated as a stochastic MIP problem, known as an L-shaped optimization problem [5]. An L-shaped problem is formulated as two-stage stochastic optimization. This formulation involves here-and-now and wait-and-see decision variables. It is a two-stage optimization problem with a single integer recourse decision in the second stage. The name L-shaped comes from the shape of the SCUC feasible region when represented graphically. This type of problem is commonly encountered in decision-making under uncertainty, where a decision-maker may face different scenarios or outcomes with associated probabilities. Due to the block structure, the Lshaped two-stage stochastic program is an effective method for SCUC [6]. L-shaped optimization problems can be solved using specialized techniques such as Benders decomposition

This work was supported by the National Science Foundation under Grant ECCS-1944752

Fouad hasan and A. Kargarian are with the Department of Electrical and Computer Engineering, Louisiana State University, Baton Rouge, LA 70803 USA (e-mail: fhasan1@lsu.edu, kargarian@lsu.edu).

(BD). Although Benders decomposition solves L-shaped problems, it suffers from exponential worst-case computational complexity. This is due, in part, to the large number of successively added cuts over iterations, which causes the size of Benders' master problem to grow excessively. Moreover, the master problem takes over 90% of the time required to implement Benders decomposition [7]. Also, as noted in [8], not every extreme point in the feasible region of subproblems contributes equally to limiting the optimal solution to the master problem. This implies that a significant number of Benders cuts may not be tight enough at the final optimal solution. Consequently, these non-useful Benders cuts can make solving large-scale integer programs challenging.

Various mathematical and heuristic approaches have been presented in the literature to enhance Benders decomposition performance. This study aims to accelerate the convergence of Benders decomposition by taking advantage of machine learning techniques.

B. Literature Review

It has been more than 50 years since the development of the Benders decomposition algorithm by J. Bender (1962) [9]. The algorithm is designed to address complicating variables that, when temporarily fixed, simplify the problem significantly. It distributes the computational load between a master problem and a subproblem (SP). Benders decomposition has proven successful in various fields, including planning and scheduling, healthcare, transportation, telecommunications, energy and resource management, and chemical process design [6]. The primary application of Benders decomposition is initially focused on solving MIP problems. Once integer variables are fixed, the problem is converted into a continuous linear program, which can develop cuts using standard duality theory. Many enhancements have been made to extend Benders' applicability to a wider range of problems. As a result, Benders decomposition has been widely used to solve linear, nonlinear, integer, stochastic, multi-stage, and bilevel optimization problems [10]. The traditional implementation of Benders decomposition can be computationally expensive, time-consuming, and memory-intensive, with problems such as poor feasibility and optimality cuts, ineffective early iterations, and the zigzagging behavior of primal solutions [7]. Researchers have explored various strategies to speed up the convergence and reduce the number of iterations and time spent on each iteration. The master problem is usually solved using branch-and-bound, with the simplex approach used to solve the subproblem. However, a significant number of cuts generated do not contribute to convergence, leading to memory occupation [8]. To address this problem, improvement criteria are proposed to ensure that new and useful cuts are included in the master problem [11]. Additionally, researchers have observed several orders of improvement when using constraint programming to solve the master problem [12]. Column generation has been introduced to handle specific

structures more effectively and achieve tighter constraints at the root node of the branch-and-bound tree [13], [14]. For large subproblems, decomposition, parallelization, and column generation have been used to reduce overall solution time. By adding valid inequalities to the master, one can significantly reduce the number of generated cuts and solution time [15]. Moreover, clustering subproblems can decrease the number of iterations [16]. Recently, researchers have been exploring the use of machine learning (ML) to alleviate the computational complexity of complex problems [17], [18]. ML-based algorithms can be categorized into three types: end-to-end learning (with label), unsupervised/reinforcement learning (no ground truth), and algorithm-specific hybrid learning, which leverages the specific structure of the target problem to accelerate the optimization process. For instance, a support vector machine is used in [19] to construct a cut classifier that identifies valuable cuts in each Benders iteration, thus reducing the size of the master problem and shortening the solution time. Another strategy is to use the Lagrange multipliers of the Benders subproblem to aggregate the optimality cuts [20]. In [21], a cut regressor and a cut classifier are used, and five features are designed to characterize the generated cuts [22]. Solving twostage stochastic SCUC with Benders decomposition is still computationally challenging and needs further investigation. Also, despite machine learning advantages, the literature lack ML-based Benders approaches to accelerate two-stage stochastic SCUC.

C. Contribution

This paper presents combined machine learning and modelbased variants of Benders decomposition for solving twostage stochastic SCUC. The proposed approaches form tighter cuts and a truncated, less computationally expensive Benders master problem. A regressor is trained to predict proxy variables corresponding to subproblems and create a tighter master problem feasible region. After carrying out each Benders iteration, useful cuts are identified using a criterion and a classification learner. A truncated master problem is created by adding useful cuts to the model and discarding nonuseful cuts. Combining the regressor and classifier reduces the computational cost and memory usage of the Benders decomposition for solving two-stage stochastic SCUC. The Benders iterative loop inherently ensures the feasibility and optimality of results. The three enhanced Benders approaches are tested on various test systems, and the results are analyzed.

D. Paper Organization

The rest of the paper is structured as follows. The problem formulation is given in Section II. The proposed enhanced Benders approaches are presented in Section III. The numerical simulations are discussed in Section IV, and concluding remarks are provided in Section V.

II. TWO-STAGE STOCHASTIC UNIT COMMITMENT

The considered problem is a two-stage stochastic unit commitment with respect to demand uncertainty. Benders decomposition is used to solve the problem.

A. Problem Formulation

The problem is formulated in (1), which includes two sets of variables pertaining to here-and-now decisions and wait-and-see decisions [23]. On/off status of units are here-and-now variables, and generation dispatches are wait-and-see

variables. The first term of (1a) is the first-stage startup and shutdown costs, and the second is the second-stage generation dispatch costs. The first stage constraints are (1b) - (1i), which are scenario-independent unit commitment constraints. The second-stage scenario-dependent operational constraints are formulated in (1j)-(1n). N-1 security constraints are included in the model with index c.

$$\min \sum_{t} \sum_{g} (SU_g y_{gt} + SD_g z_{gt}) + \sum_{\omega} \pi_{\omega} \sum_{t} \sum_{g} f(p_{gt\omega})$$
(1a)

c f

$$y_{qt} - z_{qt} = u_{qt} - u_{q(t-1)} \quad \forall g, \forall t \tag{1b}$$

$$y_{gt} + z_{gt} \le 1 \quad \forall g, \forall t$$
 (1c)

$$\sum_{t=1}^{UT_g} (1 - u_{gt}) = 0 \quad \forall g$$
 (1d)

$$\sum_{\tau=t}^{t+T_g^{on}-1} u_{g\tau} \geq T_g^{on} y_{gt} \ \forall g, t = UT_g+1, UT_g+2, ...T-T_g^{on}+1 \tag{1e}$$

$$\sum_{\tau=t}^{T} (u_{g\tau} - y_{gt}) \ge 0 \quad \forall g, t = T - T_g^{on} + 2, ...T$$
 (1f)

$$\sum_{t=1}^{DT_g} u_{gt} = 0 \quad \forall g \tag{1g}$$

$$\sum_{\tau=t}^{t+T_g^{off}-1} (1-u_{g\tau}) \ge T_g^{off} z_{gt} \ \forall g, t = DT_g+1, DT_g+2, ...T-T_g^{off}+1 \tag{1h}$$

$$\sum_{\tau=t}^{T} (1 - u_{g\tau} - z_{gt}) \ge 0 \quad \forall g, t = T - T_g^{off} + 2, ...T \quad (1i)$$

 $p_g^{min}u_{gt} \le p_{gt\omega c} \le p_g^{max}u_{gt} \quad \forall g, \forall t$ (1j)

$$p_{gt\omega c} - p_{g(t-1)\omega c} \le RU_g(1 - y_{gt}) + P_g^{min}y_{gt} \quad \forall g, \forall t \quad (11)$$

$$p_{g(t-1)\omega c} - p_{gt\omega c} \le RD_g(1 - z_{gt}) + P_g^{min} z_{gt} \quad \forall g, \forall t \quad (11)$$

$$\sum_{q} p_{gt\omega c} = D_t \ \forall t \tag{1m}$$

$$-PL^{max} \le SF_c P_{t\omega c}^{inj} \le PL^{max} \quad \forall t \tag{1n}$$

 $\forall \omega \in \Omega, \forall c$

B. Benders Decomposition

The computational complexity of mixed-integer program (1) increases with the increasing size of the network, number of demand scenarios, and number of contingencies. Benders decomposition is suitable for solving MIP problems with a block structure over contingencies and uncertainty scenarios [6]. We use a multi-cut variant of Benders decomposition that converges faster than the classical single-cut approach [20], [24]. The multi-cut technique decomposes a problem into multiple subproblems and generates multiple cuts in each iteration. This multi-cut generation improves convergence performance [25]. Problem (1) is decomposed into a master problem and n subproblems. The master problem (MP) at iteration k is formulated in (2a)-(2c).

$$\min_{x} \sum_{t} \sum_{g} (SU_{g}y_{gt} + SD_{g}z_{gt}) + \sum_{\omega} \alpha_{\omega}$$
 (2a)

s.t.

$$(1b) - (1i) \tag{2b}$$

$$\bigcup_{i=1}^{i=k-2} h_{\beta}^{n}(u_{at}, \alpha_{\omega}) \tag{2c}$$

$$h_{\beta}^{k-1}(u_{gt}, \alpha_{\omega}) : \alpha_{\omega} \ge J_{SP, \omega}^{k-1} + \sum_{t} \sum_{g} \lambda_{gt\omega}^{k-1}(u_{gt} - u_{gt}^{k-1}) \quad \forall \omega$$

$$\alpha_{\omega} \ge \alpha_{\omega}^{min} \quad \forall \omega \tag{2d}$$

$$x \in \{u_{gt}, y_{gt}, z_{gt}, \alpha_{\omega}\}$$

The objective function (2a) consists of startup and shutdown costs and a term as a proxy for the subproblems' objective function. Equation (2c) is the accumulation of all cuts generated up to iteration k-2, (2d) denotes Benders cuts generated at iteration k-1, and (2e) defines the bound on proxy variables where α_{ω}^{min} is set as a large negative constant.

Benders subproblem ω (SP_{ω}) is formulated in (3a)- (3e). To gain extra computational efficiency, we use an always-feasible subproblem model [26]. This model includes nonnegative slack variables ν and γ to prevent (SP_{ω}) infeasibility. Auxiliary variables ν and γ relax generation capacity constraints.

$$\min_{x} \sum_{t} \sum_{g} \sum_{\omega} (\pi_{\omega} f(p_{gt\omega}) + \nu_{(gt\omega)}) + \sum_{t} \sum_{\omega} \gamma_{t\omega} \quad (3a)$$

s.t.

}

$$u_{gt} = u_{gt}^* \tag{3b}$$

$$(1j) - (1n) \tag{3c}$$

$$p_q^{min}u_{gt} \le p_{gt\omega c} \le p_q^{max}u_{gt} + \nu_{(gt\omega)} + \gamma_{t\omega} \quad \forall g, \forall t \quad (3d)$$

$$\nu_{(gt\omega)} \ge 0, \gamma_{t\omega} \ge 0$$
 (3e)

$$x \in \{p_{at\omega c}, \nu_{(at\omega)}, \gamma_{t\omega}\}$$

The objective function (3a) consists of generation costs and constraint violation penalty modeled by ν_{gtw} and γ_{tw} . Constraint (3b) sets the first stage decisions, i.e., generator unit status, as fixed values received from the master problem. Inequalities (3e) set bound on auxiliary variables. The MP and SPs are solved iteratively until convergence tolerance ε is smaller than a predetermined threshold.

```
    Algorithm I Two-stage stochastic SCUC with multi-cut Benders

    1. Initialize convergence tolerance \varepsilon, set iteration index k = 0, and set \alpha_{\omega}^{min}

    2. while \varepsilon > tolerance

    3. k = k + 1

    4. Solve master problem (2) to obtain u_{gt}^k

    5. for \omega = 1:n

    6. Solve SP_{\omega} (3) to obtain J_{SP,\omega}^k and \lambda_{at\omega}^k
```

7. **end for**8. Calculate convergence tolerance $\varepsilon = \frac{UB - LB}{abs(LB)}$

9. Form new cuts as $\alpha_{\omega} \ge J_{SP,\omega}^k + \sum_{g} \sum_{t} \lambda_{gt\omega}^k \left(u_{gt} - u_{gt}^k \right)$ $\forall \omega$ and go to Step 2

10. end while

$$\epsilon = \frac{UB - LB}{abs(LB)} \tag{4}$$

where

$$LB = J_{MP}, \quad UB = J_{SP} + J_{MP} - \sum_{\omega} \alpha_{\omega}$$

The two-stage stochastic SCUC with multi-cut Benders is summarized in Algorithm I.

III. PROPOSED LEARNING ASSISTED BENDERS DECOMPOSITION

While multi-cut Benders outperforms single-cut Benders in terms of convergence performance, its memory usage and computational overhead still need enhancement. We present three learning-aided approaches to reduce the computational costs of multi-cut Benders decomposition for solving two-stage stochastic SCUC. We mainly focus on the master problem, which has higher computational costs than subproblems.

A. Regression-assisted Benders (R-Benders)

The first proposed approach uses a combination of a regressor and cut filtering. The regressor forms a tighter bound for the master problem at the root node starting from k=1. We predict the optimal subproblem proxy variables α_{ω}^* , i.e., α_{ω} upon the convergence of Algorithm I. Knowing α_{ω}^* , or even a good approximation, makes inequalities (2c), (2d), and (2e) tighter and the lower bound LB closer to the optimal subproblem cost. The cut filtering approach identifies and drops non-useful cuts during each subsequent Benders iteration to reduce the size of the master problem. Non-useful cuts are cuts with no impact on the master problem feasible region.

1) SP Objective Proxy Prediction: In the multi-cut Benders formulation (2), α_{ω} is a variable and α_{ω}^{min} is a bound where $\alpha_{\omega} \geq \alpha_{\omega}^{min}$ constitutes a master problem constraint. The value of α_{ω} increases after each iteration and reaches its optimal value α_{ω}^{*} upon convergence. The value of α_{ω}^{min} can be selected by analyzing the physical and economic aspects of the SCUC problem. Setting a suitable α_{ω}^{min} reduces the search space and enhances Benders convergence performance. An ideal case is to set α_{ω}^{min} corresponding to each subproblem ω as the optimal value of α_{ω} instead of using a large negative value. Fig. 1 shows the advantage of using optimal α_{ω}^{*} . The master problem objective reaches its optimal value in the very initial iterations. We use machine learning to predict α_{ω}^{*} for each subproblem ω by reading power demand before implementing multi-cut Benders Algorithm I.

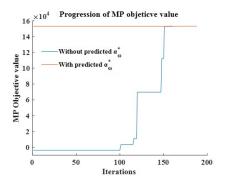


Fig. 1. Benders convergence using $\alpha_{\omega}^* \forall \omega$ in inequality (2b).

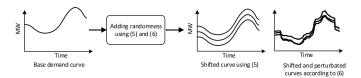


Fig. 2. Demand scenario generation.

2) Training Dataset Preparation: Electricity demand scenarios are input to two-stage SCUC. We have observed that α^*_{ω} are correlated with system demand. We use demand information as the input to a supervised learning model whose output is α^*_{ω} . To account for potential system operation scenarios in the training phase, we generate a set of daily load profile samples $D^s \in \mathbb{R}^{s \times 1}$ according to (5). To follow the concept of demand uncertainty modeling in the two-stage SCUC problem, each D^s contains n demand profile scenarios generated by (6).

$$D^s = D_{base}[\eta_s^L + \eta_s(\eta_s^U - \eta_s^L)] \quad \forall s \tag{5}$$

$$D_{\omega}^{s} = D^{s} [\eta_{\omega}^{L} + \eta_{\omega} (\eta_{\omega}^{U} - \eta_{\omega}^{L})] \quad \forall \omega, \forall s$$
 (6)

where $D_{\omega}^{s} \in \mathbb{R}^{s \times 1}$, and $\eta_{s}\{\cdot\}$ and $\eta_{\omega}\{\cdot\}$ follows a uniform distribution between 0 and 1. Indices for samples and subsamples are denoted by s and ω . Two stages of randomness are carried out to consider a range of realistic operational conditions. The load point is shifted using the random parameter $\eta_{s}\{\cdot\}$ to model its daily and seasonal volatility within the range $[\eta_{s}^{U}, \eta_{s}^{L}]$. η_{s}^{U} and η_{s}^{L} can be determined by historical data and load growth projection or to the point when any further increase or decrease renders unit commitment infeasible. The uncertainty in subsamples is modeled using the random parameter $\eta_{\omega}\{\cdot\}$ within a specified range $[\eta_{\omega}^{U}, \eta_{\omega}^{L}]$. Subsamples model the hourly load randomness by multiplying D^{s} by random parameters and creating D_{ω}^{s} . Fig. 2 illustrates the load scenario generation.

For a given system, Algorithm I is executed for each demand profile sample D^s . Optimal values of proxy variables upon Benders convergence are labeled as α^{s*} and stored. Expressions (7) and (8) are, respectively, the input and target of the supervised learning model.

Input sample:
$$D^s = [D_1^s, D_2^s, D_3^s, ..., D_n^s]^T \ \forall s$$
 (7)

Target sample:
$$\alpha^{s*} = [\alpha_1^{s*}, \alpha_2^{s*}, \alpha_3^{s*}, ..., \alpha_n^{s*}]^T \ \forall s \ (8)$$

TABLE I HYPERPARAMETERS OF THE LEARNER

NN regressor	Hidden	layer=1,	Activation = ReLU, Loss func-
	Batch size=300 500		tion = MSE Optimizer = Adam

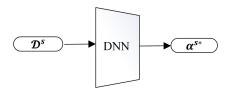


Fig. 3. NN regressor reads demand profiles and predicts proxy variables α .

3) Supervised Learning Strategy: We use neural networks (NN), an efficient tool to capture the complexity and nonlinearity of a function by utilizing various activation functions. The regressor uses a fully connected NN with mini-batch gradient descent and Rectified Linear Units (ReLU) activation functions for hidden layers. The loss function is the mean squared error (MSE).

$$MSE = \frac{\sum_{n} (x_n - \hat{x_n})^2}{n} \tag{9}$$

The Adam optimizer determines optimal weights and trains the learning model. Various batch sizes, epochs, and layer counts are tested to determine the best architecture. A single hidden layer is selected to have a minimalistic model. Table I shows the architecture and hyperparameters used in this study. Although more sophisticated structures may improve the results, we obtained promising outcomes with this minimalistic architecture. Fig. 3 illustrates a conceptual schematic of the learner model.

4) Data Scaling: Data normalization and standardization are crucial before training a learner. The data samples are normalized by (10). This process enhances prediction accuracy and numerical stability.

$$x_{normalized} = \frac{x_n - x_{min}}{x_{max} - x_{min}} \tag{10}$$

5) Useful Cut Identification: The master problem still retains all cuts generated after carrying out each Benders iteration. As a result, the master problem size grows at every iteration, calling for a considerable computational memory requirement. In most cases, a subset of Benders cuts generated at each iteration contains the necessary information to build the feasible search space of the master problem. Currently, a lack of practical and systematic approach exists for classifying useful and non-useful cuts for large-scale problems [10]. Several features are suggested in [19], [21], such as cut violation, cut depth, cut order, and cut producing scenario, to identify approximately useful cuts.

Our idea of useful cut identification is based on Fig. 4, which shows an example of the master problem objective value (or lower bound) improvement with cumulatively added cuts. It can be observed that all cuts are not equally contributing to the lower bound and optimality gap improvement. Several cuts provide a positive increase in the lower bound. Such cuts can be classified as useful cuts. Other cuts do not contribute to the lower bound improvement and can be classified as non-useful. Our observation and experiment show that the numerical difference between cut values $\psi(u_{gt}^k)$ and proxy values α_ω^k can

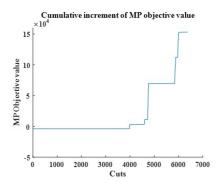


Fig. 4. Benders lower bound improvement with cuts added cumulatively; a case of IEEE 118-bus system.

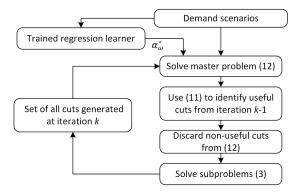


Fig. 5. R-Benders flowchart.

be used for useful cut identification. If an inequality constraint is satisfied as equality upon solving optimization, it is typically referred to as a binding or strictly useful constraint. Thus, if α_ω^k - $\psi(u_{gt}^k)=0$, the cut is useful.

However, many other cuts that are not satisfied as equality may also contain the necessary information to form the master problem feasible region. Such cuts should also be considered useful. As an inequality constraint approaches equality, its importance in forming a feasible region is expected to increase. Consider a demand profile sample D^s . To capture all cuts that contain necessary information at each iteration k, we label cuts as "useful" if the difference between α_ω^k and $\psi(u_{gt}^k)$ is less than a threshold δ , which is selected through experimental observation. We have tested various cases and observed that δ within the range [0.1, 100] works well for the studied cases. Once all cuts are generated at iteration k-1, we add them to MP and solve the updated MP at iteration k to obtain u_{at}^k and α_{ω}^{k} . We check (11) to identify useful cuts generated at iteration k-1. Cuts that satisfy (11) contain necessary information and are useful. We repeat this process at each iteration k to detect useful cuts generated at iteration k-1. Non-useful cuts are dropped from MP before moving to iteration k+1.

$$h_{\beta}^{k-1}(u_{gt}, \alpha_{\omega})^{\phi} : |\alpha_{\omega}^{k} - \psi(u_{gt}^{k})| \le \delta$$
 (11)

where $\psi(u_{gt}^k)$ is the numerical value of the right-hand side of (2d).

$$\psi(u_{gt}^k) = J_{SP,\omega}^{k-1} + \sum_t \sum_g \lambda_{gt\omega}^{k-1} (u_{gt} - u_{gt}^{k-1})$$

6) Avoiding Information Loss: Selecting a small δ results in labeling fewer cuts as useful. A large δ yields keeping more cuts in the master problem, and thus less computational cost reduction is gained. Depending on the value of hyperparameter δ , we have observed cases with no cut labeled as useful at a few iterations. Retaining at least one cut from each iteration is crucial; otherwise, the proposed R-Benders may not converge due to loss of information from that particular iteration. To prevent this, we keep several cuts generated from high-load scenarios at each iteration, most of which may already be on the useful cut list. The number of retained cuts is determined through experimentation.

7) R-Benders MP Formulation: The R-Benders master problem at iteration k is as follows:

$$\min_{x} \sum_{t} \sum_{g} (SU_{g}y_{gt} + SD_{g}z_{gt}) + \sum_{\omega} \alpha_{\omega}$$
 (12a)

s.t.

$$(1b) - (1i)$$
 (12b)

$$\bigcup_{i=1}^{i=k-2} h_{\beta}^{n}(u_{gt}, \alpha_{\omega})^{\psi}$$
 (12c)

$$h_{\beta}^{k-1}(u_{gt},\alpha_{\omega}):\alpha_{\omega}\geq J_{SP,\omega}^{k-1}+\sum_{t}\sum_{q}\lambda_{gt\omega}^{k-1}(u_{gt}-u_{gt}^{k-1})\ \forall \omega$$

$$\alpha_{\omega} \ge \alpha_{\eta} \alpha_{\omega}^{min} \quad \forall \omega \tag{12d}$$

$$x \in \{u_{gt}, y_{gt}, z_{gt}, \alpha_{\omega}\}$$

where (12c) denotes the list of all useful cuts detected until iteration k-2. (12d) includes all cuts generated at iteration k-1. The value of α_ω successively progresses to reach the optimal value. To maintain the quality of the lower bound solution, we should ensure $\alpha_\omega \geq \alpha_\omega^*$. Given the possibility of machine learning error, the regressor predictions α_ω^* are reduced by a factor α_η and $\alpha_\omega \geq \alpha_\omega^*$ is replaced with $\alpha_\omega \geq \alpha_\eta \alpha_\omega^*$. The reduction factor α_η is determined by analyzing the regressor prediction error. The R-Benders approach is summarized in Algorithm II and Fig. 5. Although a classification learning model can be trained to identify useful cuts (as in the C-Bender approach presented in Section III.B), the R-Benders approach does not use any classifier with the cost of filtering cuts with one iteration delay. The only learning model in R-Benders is a regressor to predict α_ω^* .

B. Classification-based Cut Identification (C-Benders)

We use a supervised classification learner to identify useful cuts before solving the master problem at each iteration k, instead of using criterion (11). Fig. 6 conceptualizes the proposed C-Benders where dotted lines denote non-useful cuts and solid lines denote useful cuts. The input to the classifier is cuts generated at iteration k, and its output is cut labels. Only useful cuts are kept; the rest are omitted from the master problem feasible region. A neural network is trained with a suitable loss function, such as F-score. The model architecture and hyperparameters are given in Table I.

To prepare a training dataset, all cuts must be labeled as useful or non-useful. We implement Benders for each demand profile sample and collect all cuts upon convergence. We restart the Benders only to solve the master problem while ignoring subproblems as we have already stored cuts. At each iteration k, we add to the master problem one of the stored cuts and measure its contribution to lower bound improvement. We repeat this procedure by cumulatively adding cuts

Algorithm II Proposed R-Benders decomposition Initialize convergence tolerance ε , set iteration index k=0, Feed predicted demand scenarios to the regressor to predict 3. while ε > tolerance 4. k = k + 1Solve Benders master problem (12) to obtain α_{ω}^{k} and u_{at}^{k} 5 If $k \ge 2$ 6. 7. Use Benders master problem solution obtained to identify useful cuts 8. Drop non-useful cuts from master problem 9 end if 10. for $\omega = 1$: n11. Solve SP_{ω} (3) to **obtain** $J_{SP,\omega}^{k}$ and $\lambda_{gt\omega}^{k}$ 12. end for Calculate convergence tolerance $\varepsilon = \frac{UB - LB}{abs(LB)}$ 13. 14 Form new cuts as $\alpha_{\omega} \ge J_{SP,\omega}^k + \sum_{g} \sum_{t} \lambda_{gt\omega}^k \left(u_{gt} - u_{gt}^k \right)$ $\forall \omega$ and go to Step 3 15. end while

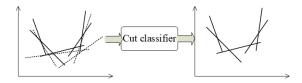


Fig. 6. Useful cut identification with a classifier.

one after another to label all cuts of iteration k depending on their contribution to the lower bound improvement. We repeat this process for all Benders iterations. One alternative is to form the master problem with all cuts first and then remove cuts one by one to observe their impact on the lower bound. Generating a labeled dataset for C-Benders is computationally expensive, particularly for multi-cut Benders decomposition. The number of Benders iterations is relatively high for the SCUC problem with intertemporal constraints, leading to a multiplicatively larger number of cuts that should be individually checked. The number of cuts generated at each iteration depends on the number of demand scenarios. This is another factor that increases the computational cost of cut labeling. For instance, consider solving the 118-bus system with a 4-hour commitment horizon and a load profile sample with 40 stochastic scenarios. This case takes 289 iterations to converge with a less than 1% gap. The total number of cuts generated for each sample is $40 \times 289 = 11,560$. To determine the label of each cut, we must solve the master problem 11,560 times. Generating the training dataset for C-Benders is computationally expensive, while R-Benders does not need such dataset generation. However, C-Benders provides more accurate cut labels than R-Benders, which estimates labels using (11). Fig. 7 shows the C-Benders flowchart. The master problem is formulated in (12a)-(12d), and (2e). Filtered cuts in (12c) are identified using the classifier. The training process is offline and needs to be performed only once. The overall accuracy of the learning-aided cut classification is slightly better than the non-learning cut classification of R-Benders.

C. Combined Classifier-Regressor Benders (CR-Benders)

This approach (Fig. 8) is a combination of C-Benders and R-Benders. CR-Benders uses a regressor to predict subproblem

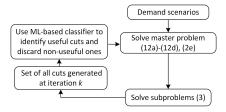


Fig. 7. C-Benders flowchart.

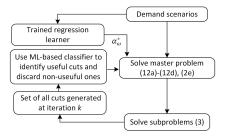


Fig. 8. CR-Benders flowchart.

proxy variables α_{ω}^* and a classifier for cut filtering before starting each iteration k. This approach is C-Benders equipped with α_{ω}^* prediction capability.

D. Subproblem Acceleration

Two non-binding constraint removal strategies can be used to accelerate subproblem solutions, particularly for large SCUC problems. In the first learning-aided strategy, two classifiers can be trained to identify inactive (aka non-binding) network and generation ramp constraints at each Benders iteration and remove them from the model [27]. Another more straightforward yet less efficient strategy is to solve a relaxed subproblem without network constraints. The relaxed subproblem solution and shift factors are used to check network constraints. The violated constraints are added to the relaxed subproblem, which is repeated until all network constraints are satisfied [28]. These two strategies reduce computation time and memory usage. We have used the second strategy.

IV. NUMERICAL RESULT

The effectiveness of the proposed algorithms is evaluated in comparison to conventional multi-cut Benders decomposition. The YALMIP toolbox and CPLEX are used to implement Benders [29], and PyTorch is used to construct neural network models. Simulations are conducted on a computer with an Intel(R) Xeon(R) 2.10 GHz CPU and 512 GB of RAM.

A. Test Systems and Data Preparation

Three test systems with various scheduling horizons are used, including the IEEE 24-bus, 118-bus, and 1354-bus systems [30]. The 24-bus system includes ten generators and 34 transmission lines. The 118-bus system consists of 54 generators and 186 lines. The 1354-bus system has 270 generators and 1991 branches. Each case includes 40 equiprobable load profile scenarios. Table II shows the range of load perturbation with respect to base case demand values to incorporate daily and hourly load uncertainty. The convergence tolerance is set to 1% for all cases.

TABLE II PARAMETER RANGE FOR DEMAND DATASET GENERATION

System	Load	
	$[\eta_s^L,\eta_s^U]$	$[\eta_{\omega}^L,\eta_{\omega}^U]$
24-bus	70%-130%	95%-105%
118-bus	70%-130%	95%-105%
1354-bus	70%-110%	95%-105%

TABLE III
COMPARISON OF TOTAL CUTS AND USEFUL CUTS

Test case	Average number of total cuts	Average number of useful cuts	% of useful cutFig.
24-bus,1h	224	103	46%
24-bus,12h	1872	220	12%
118-bus,1h	308	109	35%
118-bus,3h	2280	288	13%
1354-bus,2h	5520	795	15%

B. Useful Cut Statistics

Table III shows the average number of total cuts and useful cuts for different systems where h stands for horizons. To determine the usefulness of a cut, we have calculated the contribution of each cut to improving the objective value of the master problem. A cut is useful if it results in a non-zero increase in the objective value. The percentage of the number of useful cuts to total cuts reduces as the size of the system and the number of scheduling horizons increase. For instance, for the IEEE 118-bus system with three time periods, only 13% of cuts are labeled as useful, and the rest of 87% are non-useful. Filtering non-useful cuts reduces the computational burden of the master problem significantly.

C. Master Problem Runtime and Solution Quality

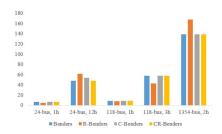
Table IV reports the average solver time of the master problem and the percentage of time improvement obtained by the three proposed approaches as compared to the conventional multi-cut Benders. During each iteration of the proposed approaches, we tackle a significantly smaller problem, resulting in considerable time savings. All three proposed approaches outperform conventional Benders. The time saving becomes more significant as the size of the optimization model increases. For instance, R-Benders reduces the master problem solution time from 236 seconds to 30 for the 118-bus system, an 87% improvement. The time-saving of CR-Benders is more than R-Benders and C-Benders.

The average number of Benders iterations is shown in Fig. 9. The three proposed approaches, particularly C-Benders and CR-Benders, take roughly the same number of iterations as conventional Benders decomposition. A comparison of the solution time and the number of iterations, as well as the number of useful cuts from Table III, show that R-Benders, C-Benders, and CR-Benders can capture the necessary information to build a computationally less expensive master problem.

We use a CostGap index to assess the solution quality. The CostGap index measures the difference between the objective

TABLE IV
AVERAGE SOLVER TIME (AND IMPROVEMENT PERCENTAGE)
COMPARISON IN SECONDS

System	Benders	R-Benders	C-Benders	CR-Benders
24-bus,1h	2.1	1.4 (33%)	0.73 (65%)	0.66 (69%)
24-bus,12h	37.7	32.9 (13%)	9.7 (74%)	9.4 (75%)
118-bus,1h	2.8	2.6 (7%)	2.5 (11%)	2.5 (11%)
118-bus,3h	236	30 (87%)	57 (76%)	24 (90%)
1354-bus,2h	110	74 (33%)	65 (41%)	62 (44%)



9. Average number of iterations.

TABLE V COSTGAP INDEX (%) R-Benders CR-Benders System C-Benders 24-bus,1h U 0.13 0.07 24-bus,12h 0.03 0.05 0.8 118-bus,1h 0.6 0.010.01118-bus,3h 0 0.02 0.02 1354-bus,2h

values generated by the proposed approaches f^p and the conventional Benders $f^B D$.

$$CostGap\% = \frac{|f^P - f^{BD}|}{f_{BD}} \times 100 \tag{13}$$

Table V presents the average CostGap index for all cases. The average cost gap for all test cases is negligible. For instance, the gap is less than 0.02% for the 118-bus system with a 3-hour horizon. We have observed several cases for which the proposed approaches obtain even a better solution than the conventional Benders. This is due to a warm start with stronger cuts in the first iteration.

D. Memory Usage

The memory usage of the proposed approach is compared with the conventional Benders. The memory occupied to form the constraint set from Benders cuts is illustrated in Fig. 10. The memory requirement reduces significantly by filtering out non-useful cuts from the model. For example, Fig. 10.d shows that for the 118-bus system with 3h horizon, the memory usage of CR-Benders is 87.5% less than that of the conventional Benders decomposition. Fig. 11 demonstrates the number of cuts added to the master problem per iteration. The slope of the increment for the conventional Benders is steeper than that of RC-Benders.

E. Non-convergent case analysis

In specific situations, the conventional Benders decomposition (BD) technique struggles to achieve convergence within

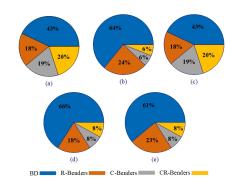


Fig. 10. Memory usage comparison. A) 24-bus (1h), b) 24-bus (12h), c) 118-bus (1h), d) 118-bus (3h), and e) 1354-bus (2h).

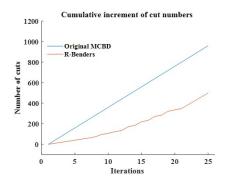


Fig. 11. Cumulative increment of the number of cuts for IEEE 118-bus 3h

the allocated timeframe, even in the presence of an optimal solution. Through our experiments, we have demonstrated that the utilization of a predicted subproblem proxy can effectively address these scenarios and enhance the duality gap. We focused on two particular scenarios involving a 118-bus system and a 3-hour horizon, where the original MCBD approach failed to converge within 400 iterations. By employing the Rbenders method, we observed an average convergence within 59 iterations for these same scenarios. It is evident that the incorporation of the proxy value yields superior performance.

V. CONCLUSION

The proposed algorithms aim to reduce the computational costs of two-stage SCUC using machine learning-based warmstart and cut filtering. The proposed algorithms provide guaranteed feasible and near-optimal solutions. Conventional Benders decomposition suffers from slow convergence and might not reach an optimal point within a specified time. Not all cuts generated through Benders iterations contain useful information to form the feasible region of the master problem. Removing non-useful cuts and forming stronger cuts enhance Benders decomposition performance. We trained a regressor to predict subproblem objective proxy variables for the master problem. The predicted values form the first set of cuts in the first Benders iteration. After solving the first iteration and onward, cuts are filtered by comparing the numerical value of cuts and the numerical value of proxy variables obtained from that particular iteration. A classifier is trained to identify useful cuts automatically after each Benders iteration. The proposed approaches are applicable to many MIP problems. Simulation results on various test systems show CR-Benders outperform R-Benders and C-Benders in terms of solution time and memory saving. On average, CR-Benders leads to 58% time saving and 77% memory saving, which is higher than the other two approaches. While C-Benders and CR-Benders work better than R-Benders, they need a computationally expensive dataset generation for classification learning training. This procedure is, however, offline and needs to be implemented once.

REFERENCES

- A. J. Conejo, L. Baringo, A. J. Conejo, and L. Baringo, "Unit commit-ment and economic dispatch," *Power System Operations*, pp. 197–232,
- T. Yang, D. Wu, Y. Sun, and J. Lian, "Minimum-time consensusbased approach for power system applications," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 2, pp. 1318–1328, 2016.
- Y. Du, Y. Li, C. Duan, H. B. Gooi, and L. Jiang, "Adjustable uncertainty set constrained unit commitment with operation risk reduced through demand response," IEEE Transactions on Industrial Informatics, vol. 17, no. 2, pp. 1154-1165, 2021.

- [4] W. Zhang, C. Shao, B. Hu, J. Zhou, M. Cao, K. Xie, P. Siano, and W. Li, "Proactive security-constrained unit commitment against typhoon disasters: An approximate dynamic programming approach,' IEEE Transactions on Industrial Informatics, vol. 19, no. 5, pp. 7076-7087, 2023.
- [5] C. C. Carøe and J. Tind, "L-shaped decomposition of two-stage stochastic programs with integer recourse," Mathematical Programming, vol. 83, pp. 451-464, 1998.
- A. J. Conejo, E. Castillo, R. Mínguez, and R. García-Bertrand, "Decomposition in linear programming: complicating variables," Decomposition Techniques in Mathematical Programming: Engineering and Science Applications, pp. 107-139, 2006.
- T. L. Magnanti and R. T. Wong, "Accelerating benders decomposition: Algorithmic enhancement and model selection criteria," Operations research, vol. 29, no. 3, pp. 464-484, 1981.
- M. Minoux, Mathematical programming: theory and algorithms. John
- Wiley & Sons, 1986.

 J. BnnoBRs, "Partitioning procedures for solving mixed-variables programming problems '," Numerische mathematik, vol. 4, no. 1, pp. 238-252, 1962
- [10] R. Rahmaniani, T. G. Crainic, M. Gendreau, and W. Rei, "The benders decomposition algorithm: A literature review," European Journal of Operational Research, vol. 259, no. 3, pp. 801-817, 2017
- K. Holmberg, "On the convergence of cross decomposition," Mathematical Programming, vol. 47, pp. 269-296, 1990.
- A. I. Corréa, A. Langevin, and L.-M. Rousseau, "Scheduling and routing of automated guided vehicles: A hybrid approach," Computers & operations research, vol. 34, no. 6, pp. 1688–1707, 2007.
- [13] J.-F. Cordeau, G. Stojković, F. Soumis, and J. Desrosiers, "Benders decomposition for simultaneous aircraft routing and crew scheduling, Transportation science, vol. 35, no. 4, pp. 375-388, 2001.
- [14] M. I. Restrepo Ruiz, Grammar-Based Decomposition Methods for Multi-Activity Tour Scheduling. PhD thesis, École Polytechnique de Montréal,
- J. Naoum-Sawaya and S. Elhedhli, "A nested benders decomposition approach for telecommunication network planning," Naval Research Logistics (NRL), vol. 57, no. 6, pp. 519-539, 2010.
- [16] K. Brandes, Implementierung und analyse verschiedener strategien zur aggregation und disaggregation von multi-cuts im benders dekompositionsverfahren. PhD thesis, Master's thesis, Universität Paderborn, North Rhine-Westphalia, Germany, 2011
- [17] A. Paulus, M. Rolínek, V. Musil, B. Amos, and G. Martius, "Comboptnet: Fit the right np-hard problem by learning integer programming constraints," in International Conference on Machine Learning, pp. 8443-8453, PMLR, 2021.
- [18] M. Mohammadi and A. Mohammadi, "Empowering distributed solutions in renewable energy systems and grid optimization," arXiv preprint arXiv:2310.15468, 2023.
- [19] H. Jia and S. Shen, "Benders cut classification via support vector machines for solving two-stage stochastic programs," INFORMS Journal on Optimization, vol. 3, no. 3, pp. 278-297, 2021.
- [20] B. Vandenbussche, S. Delikaraoglou, I. Blanco, and G. Hug, "Datadriven adaptive benders decomposition for the stochastic unit commitment problem," arXiv preprint arXiv:1912.01039, 2019.
- [21] M. Lee, N. Ma, G. Yu, and H. Dai, "Accelerating generalized benders decomposition for wireless resource allocation," *IEEE Transactions on Wireless Communications*, vol. 20, no. 2, pp. 1233–1247, 2020.

 [22] A. Ruszczyński, "Decomposition methods," *Handbooks in operations*
- research and management science, vol. 10, pp. 141-211, 2003
- A. J. Conejo and L. Baringo, Power system operations, vol. 11. Springer,
- [24] J. R. Birge and F. V. Louveaux, "A multicut algorithm for two-stage stochastic linear programs," European Journal of Operational Research, vol. 34, no. 3, pp. 384-392, 1988.
- L. Su, L. Tang, and I. E. Grossmann, "Computational strategies for improved minlp algorithms," Computers & Chemical Engineering, vol. 75, pp. 40–48, 2015.
- A. Nasri, S. J. Kazempour, A. J. Conejo, and M. Ghandhari, "Networkconstrained ac unit commitment under uncertainty: A benders' decomposition approach," IEEE transactions on power systems, vol. 31, no. 1, pp. 412-422, 2015.
- [27] F. Hasan and A. Kargarian, "Topology-aware learning assisted branch and ramp constraints screening for dynamic economic dispatch," IEEE Transactions on Power Systems, vol. 37, no. 5, pp. 3495-3505, 2022.
- [28] D. A. Tejada-Arango, P. Sánchez-Martin, and A. Ramos, "Security constrained unit commitment using line outage distribution factors, IEEE Transactions on power systems, vol. 33, no. 1, pp. 329-337, 2017.
- J. Lofberg, "Yalmip: A toolbox for modeling and optimization in matlab," in 2004 IEEE international conference on robotics and automation (IEEE Cat. No. 04CH37508), pp. 284-289, IEEE, 2004.
- S. Babaeinejadsarookolaee, A. Birchfield, R. D. Christie, C. Coffrin, C. DeMarco, R. Diao, M. Ferris, S. Fliscounakis, S. Greene, R. Huang, et al., "The power grid library for benchmarking ac optimal power flow algorithms," arXiv preprint arXiv:1908.02788, 2019.