Capturing Unit Startup and Shutdown Uncertainties in the Real-time Commitment Process

Shengfei Yin, *Student Member, IEEE*, Yonghong Chen, *Senior Member, IEEE*, Ben Knueven, Long Zhao, Mohammad Faqiry, Anupam A. Thatte, *Member, IEEE* and Jianhui Wang, *Fellow, IEEE*

***Abstract*—Generation uncertainties, especially during the unit startup and shutdown (SU/SD) processes, pose uncertainties for the real-time market clearing process, and they are often underestimated. This paper proposes two approaches to predict generator SU/SD trajectories in the real-time operations of independent system operators or regional transmission organizations (ISO/RTOs). We first collect and pre-process raw market data from state estimation. Then we investigate two approaches to account for the uncertainty in MW of generation SU/SD in the real-time market clearing. The first is an offline approach that leverages a machine learning technique, gradient boosting tree, to effectively capture the nonlinear relationship between the SU/SD curves and selected feature maps. The offline approach works for predicting generator trajectories in the real-time Look Ahead Commitment (LAC) process, based on historical data. We also investigate an online approach using a long-short-term memory network that can learn from the last-interval error information and enhance the current prediction, potentially applicable for the real-time economic dispatch process. We validate the benefit of the proposed approach with a full-day rolling LAC framework on MISO-size test cases. The result shows that using the predicted curves could help system operators achieve better results in real-time commitment and dispatch processes.**

***Index Terms*—Generation startup and shutdown curves, real-time market operation, unit commitment, gradient boosting tree, long-short-memory network.**

I. Introduction

In its efforts to efficiently commit and dispatch resources, ISO/RTOs need to account for the startup and shutdown (SU/SD) behavior of resources. In standard market operating processes, operators send commit and decommit instructions to the generators. Generation outputs during SU/SD are metered but not projected in the market-clearing engine. In the market-clearing process, units' SU/SD MW is ignored most of the time and treated as noises. For time periods with projected large MW from SU/SD, real time operations may use manual offset to account for the estimated MW value. In addition, the MISO practice in the market clearing processes [1] only consider SU/SD binary decision for units in unit commitment constrained by a predefined SU/SD ramp rate. However, this practice does not capture the accurate MW values during SU/SD. The MW quantities below the dispatchable minimum limit are relatively small most time. But they may undermine the accuracy of market clearing results when the system has a large number of generator startups or shutdowns during net load ramping periods. The need for accurate and systematic measurements of SU/SD curves is even more important for the real-time market since it has a finer time granularity.

Research on timeseries prediction for power system applications is fruitful, but very few works investigate units' SU/SD processes as a generation uncertainty. Albeit X. Lin et al. [13] leverage random forest as a classification method to predict the SU/SD hours to reduce the number of binary variables in the unit commitment, how to let system operators accurately account for the unit SU/SD MW in real-time operations is still an open question.

Unit SU/SD curves are typical timeseries with inherent relationships with multiple potential unit-specific and environmental factors. Sample unit-specific factors include capacity size, unit ramp rate, minimum power output, and fuel type, etc. Environmental factors include ambient temperature, dewpoint, and rainfall. However, there are also differences between the features for startup curves and shutdown curves. For the startup process, units' startup behaviors could be potentially impacted by how long the unit has been offline (cold start or warm start) and the current boiler status for some units. These features would not impact the shutdown process, where the dispatch MW immediately before shutting down is a more important factor for the shape of the shutdown curve. Unlike wind and solar forecasts that have already had a mature and well-validated feature set, to the best of the authors' knowledge, there is no existing work that comprehensively discusses units' SU/SD impact factors during which their MW levels are nearly uncontrollable. Thus, the feature selection and quantitative analysis of feature importance for the SU/SD processes call for in-depth investigation. State-of-the-art feature engineering algorithms like Deep Feature Synthesis (DFS) [2] could help with such a task.

Machine learning (ML) techniques have been researched and in some cases applied for numerous power sector applications,

S. Yin, Y. Chen, L. Zhao, M. Faqiry, and A. A. Thatte are with the Midcontinent Independent System Operator (MISO), Carmel, IN, 46032 USA. Emails: {syin, ychen, lzhao, mofaqiry, athatte}@misoenergy.org.
(Corresponding Author: Yonghong Chen)

B. Knueven is with the National Renewable Energy Laboratory, Golden, CO, 15013 USA. Email:

J. Wang is with the Department of Electrical and Computer Engineering, Southern Methodist University, TX, 75206 USA. Email: jianhui@smu.edu.

including load forecast [3], renewable forecast [4], false data detection [5], state estimation [6], and system operations [7] etc.. However, there is little research on SU/SD prediction. Nevertheless, there are common core ideas across applications. ML for SU/SD behavior prediction represents a supervised learning task for timeseries based on historical data, towards which existing works have presented many efficient methods. C. Wan et al. [8] improved the existing probabilistic wind power forecast with a nonparametric prediction interval setup using quantile regression and extreme learning machine. H. Yang et al. [9] proposed a hybrid training tool for solar power forecast considering temperature and solar irradiance, which consisted of a classification stage using learning vector quantization and a regression stage using support vector regression. J. Andrade et al. [10] applied a gradient boosting tree algorithm to conduct wind speed forecast based on an enhanced spatiotemporal feature selection for timeseries data. More specifically, as the generation uncertainty is mostly timeseries, the autoregression model and long-short-term memory network (LSTM) are two mainstream methods with well-known effectiveness. J. Dowell et al. [11] employed a sparse vector autoregression method to capture very short-term spatial information for wind power forecast. M. Khodayar et al. [12] combined the graph convolutional network with LSTM to capture the spatiotemporal correlations between multiple wind farms and thus enhance the future timeseries predictions for wind speed forecast..

The ultimate goal of the SU/SD prediction is to help system operators better manage generation uncertainties in real-time operations. If we take MISO as an example, the real-time clearing includes two main components, a real-time security-constrained unit commitment (UC) routine called look-ahead commitment (LAC) and a real-time security-constrained economic dispatch (ED) routine called unit dispatch system (UDS). LAC has a fifteen-minute resolution and a three-hour horizon, while UDS in our study is a single-interval operation with a five-minute resolution. The UDS scheduling obeys the commitment decisions determined in the commitment processes. The difference between time resolutions requires different granularity and eligibility of the predicted SU/SD curves. For example, a fast-responsive unit that can startup in ten minutes will not have the startup curve in LAC, but it will have the curve in UDS, which also applies to the shutdown units. Capturing SU/SD behavior in LAC and UDS could help the low-resolution, real-time clearing software more accurately account for non-dispatchable MW. In this work, we use the MISO test cases to illustrate the efficacy of the proposed methodologies, which could be conveniently generalized to other ISO/RTOs' use cases or other research models.

We summarize the contributions of this paper as follows.

- We investigate the unique features in the unit's SU/SD behavior by employing automated feature engineering techniques. We quantitatively evaluate the importance of features for each unit and identify features with the highest impact and therefore can be used in the predictive approaches.
- We propose two predictive approaches to capture the unit SU/SD uncertainties by using state-of-the-art ML techniques. The first approach conducts a static supervised prediction using historical data and selected features. The second approach incorporates a real-time error correction to enhance predictions in finer-resolution operations.
- We validate the predicted SU/SD curves with a MISO test case by incorporating curves in the LAC and UDS co-simulation. These results could serve as references for future studies on generation SU/SD uncertainties.

II. Raw Data Preprocessing and Feature Selection

This section introduces how we conduct the preprocessing for raw data and how we select important features for later ML tasks. It is widely recognized in the ML community that regardless of the power of predictors, flawed inputs produce flawed outputs, which is the motivation of this task. Note that we only consider the SU/SD curves for conventional resources, including diesel units, combined-cycle units, steam turbines, and combustion turbines, etc. Non-dispatchable renewable resources usually can startup and shutdown very quickly and their outputs are predicted by renewable forecast.

The raw data in the ISO/RTO's database comes from the transmission state estimation and contains many noise and error measurements. This noise and error poses a substantial challenge for prediction tasks. Hence, we need first to apply data analytics methods to clean the data for SU/SD prediction purposes. It may require extra efforts to carry out individually customized data cleaning for industry-level raw data because different datasets from different measuring apparatuses may contain different degrees of noise and errors. Another preprocessing is that we filter out units that can startup and shutdown within a market interval. As the SU/SD curve predictions work for the LAC and UDS operation, if the unit can startup or shutdown within fifteen/five minutes, it would not generate a LAC/UDS curve, respectively. Preprocessing the dataset could significantly increase the prediction accuracy and reduce the computational time in the prediction task.

For data cleaning, one of the most essential steps is removing

data outliers. In this work, we apply a widely used interquartile range method as the SU/SD datasets do not follow the Gaussian distribution. It computes the bounded quantiles by solving the following integrals and generating a confidence box:

$$\int_{-\infty}^{N_1} f(x)dx = Q_1, \quad (1)$$

$$\int_{-\infty}^{N_2} f(x)dx = Q_2, \quad (2)$$

where tuple (N_1, N_2) denotes the range of quantiles and (Q_1, Q_2) denotes the selected percentiles. This method is based on the statistical assumption that the outliers occur with low probability and normal points occur with high probability, which applies to the SU/SD case.

We present the overall steps for the data preprocessing, as shown below.

1) Query the raw timeseries data from the MISO database. As LAC has a three-hour horizon, we consider SU/SD curves within two hours. Based on the MISO practice, for SU/SD predictions, we use the two hours before the startup effective time for the startup prediction and two hours after the shutdown signal is sent, respectively. Hence, each raw timeseries instance has a maximum of 24 intervals with a five-minute resolution.
2) Check the gradient of the curve. For a startup timeseries, if it has more than half consecutive negative gradients, or its curve gradient is too small for all intervals, it will be deemed a wrong measurement and removed. For the shutdown timeseries similar logic is applied.
3) Remove the outliers using the interquartile range method. In our study, we generally set $Q_1 = 0.25$ and $Q_2 = 0.75$. We also remove all duplicate measurements in this step.
4) For missing measurements in one timestamp, we impute with the median of neighborhoods. For instances with more than half missing measurements, we remove these instances.
5) If 90% timeseries instances of one unit could reach the minimum power output within five/fifteen minutes, the unit is excluded from the startup curve prediction for UDS/LAC, respectively. For the shutdown timeseries we apply similar logic.

Note that the above data cleaning criteria are developed intuitively and are customized for individual unit's raw data. We provide an illustrative example in Fig. 1 for startup data cleaning to help better understand how steps 2) and 3) work. The original data in Fig. 1. a) has some horizontal or decreasing lines, which are not aligned with the startup behavior, while the outliers in Fig. 1. b) are without indicative features and could confuse ML models. The proposed data preprocessing could effectively clean the dataset and keep the original data features.

After the data preprocessing is complete, we conduct feature selection by applying the DFS algorithm. This algorithm follows the inherent relationships between original input features and then sequentially applies mathematical functions along the relationships to create new features [2]. It is efficient

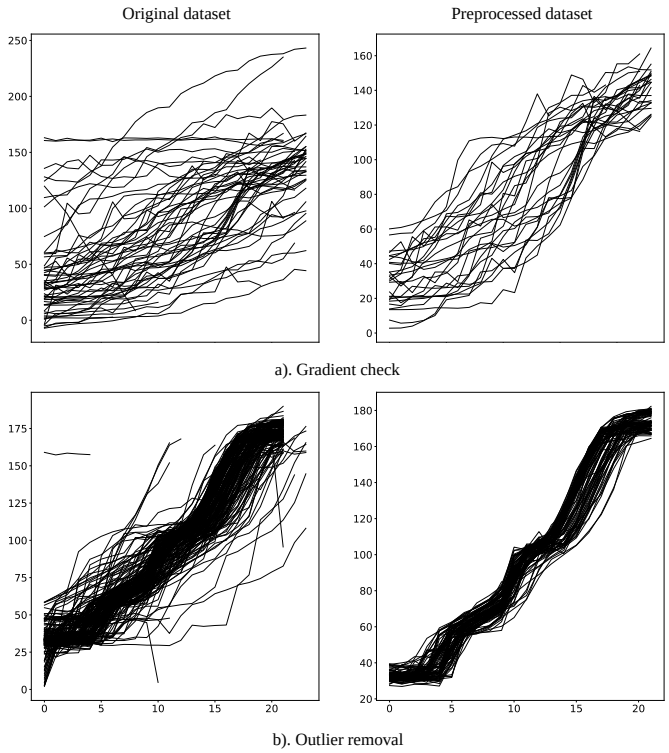


Fig. 1. Illustrative example for startup data cleaning.

when we do not know the complete list of correlated features.

In the beginning, we have an original set of selected features. Take the startup process as an example. The potential impacting features include the current time, unit ramp rate, minimum power output, fuel type, current offline time, and boiler states, etc. These features could be generalized into three parts: numeric, categorical, and time stamp. For the ML tasks described in the next section to be carried out, many regressors only work on numeric entries in the training stage, while others are also recognized to have a better performance on numeric data. Hence, for time stamp features, we apply the one-hot encoding method [14] to convert all the time stamp features to numeric features. This method works by transforming multiple data levels into a diagonal matrix-like numeric table, which gives all associated data entries a binary flag. All the preprocessed features are normalized. For categorical features, the one-hot encoding method could also be applied. However, there are too many categorical features in the SU/SD datasets, which means using one-hot encoding raises the curse of dimensionality of the datasets. Hence, we use a modified one-hot encoding for categorical features, to be discussed in Section III.

After the preparation of initial features, we apply the DFS algorithm to generate more features. The core idea of the DFS algorithm is to first build forward and backward relationships between original features, then new features could be generated by combining or reweighting the derived relationships. This process helps develop more potential impact factors from the existing feature entries by mining their inherent correlations. The generated new features will be further refined via a truncated singular value decomposition (SVD) for dimensionality reduction. More details about how to implement

this algorithm can be found in [2].

Then, we perform the feature selection according to the feature importance. It is evaluated by two metrics as shown below. We calculate the importance metrics based on a decision tree setting because we employ a customized gradient boosting tree as the main prediction tool for this study. However, the feature importance calculation metrics could be generalized to other ML models.

$$PredC = \sum_{t\in T}\sum_{\ell\in L}\left(v_{t,\ell,a} - m_{t,\ell}\right)^2 \cdot w_{t,\ell,a} + \left(v_{t,\ell,b} - m_{t,\ell}\right)^2 \cdot w_{t,\ell,b}, \quad (3)$$

$$m_{t,\ell} = \frac{v_{t,\ell,a}w_{t,\ell,a} + v_{t,\ell,b}w_{t,\ell,b}}{w_{t,\ell,a} + w_{t,\ell,b}},$$

$$LossC = |\mathcal{L}(\mathbb{E}\{\boldsymbol{v}\}) - \mathcal{L}^*| - |\mathcal{L}(\boldsymbol{v}) - \mathcal{L}^*|, \quad (4)$$

where $PredC$ and $LossC$ denote degrees of change of the prediction value and loss function value based on the change of feature values, respectively; T and L denote tree and leaf sets, respectively; a and b denote two leaf nodes of the equivalent binary tree; v and w denote the data entries and data weights, respectively; m denotes mean of the weighted data; $\mathcal{L}$ is the defined loss function and $\mathcal{L}^*$ is the observed best loss value.

For these metrics, $PredC$ is calculated based on all leaves, while $LossC$ is calculated based on the overall loss function performance. The feature importance could then be ranked based on the prediction change and loss function change when the feature changes. By filtering the feature importance, we could select important features for the later ML models to accelerate the training process while improving the accuracy.

III. Prediction Methodologies

In this section, we introduce the proposed prediction methodologies for the preprocessed SU/SD datasets.

A. Offline Approach: Categorical Boosting

For LAC and UDS operations, system operators prefer to have the SU/SD curves as steady-state input data that can be conveniently incorporated into the existing framework. In this regard, we propose leveraging the timeseries-based supervised learning to tackle this problem. Here, we use an enhanced gradient boosting tree model to generate the SU/SD curves effectively.

Gradient boosting decision tree is a boosting algorithm using ensembled decision trees. By leveraging the greedy boosting concept, stage-wise decision trees use the last-stage prediction residuals as training data to enhance the initial prediction, reducing both the bias and variance. It works the best when the data has many uncorrelated features with weak prediction potentials because the boosting tree is empowered by using ensembled small prediction models, whose predictability has been proved to beat random forests [15]. Gradient boosting tree could be used for hybrid datasets with numeric feature entries, but it is also reportedly not suitable for tasks with many categorical features [16]. Categorical features could not be directly trained with numeric features due to their mutual incomparability. Many widely applied ML algorithms such as autoregression, support vector machine, and neural networks are not the best suits for considerable categorical features, which may need additional preprocessing efforts. However, as discussed in Section II, there are many categorical features for the SU/SD curve predictions. Hence, we cannot just apply the vanilla gradient boosting tree. Instead, we enhance the decision tree by leveraging the state-of-the-art Categorical Boosting [17].

We first introduce the gradient boosting tree algorithm as follows. Consider a paired dataset $\mathcal{P} = \{(\boldsymbol{x}_i, y_i)\}_{i=1,\dots,n}$, where $\boldsymbol{x}_i = (x_i^1, \dots, x_i^k)$ is a feature vector with k features and y_i is the prediction target. In our startup/shutdown study, y_i is the numeric state estimation MW of one unit's power output. We assume that the pair $(\boldsymbol{x}_i, y_i)$ is sampled i.i.d. from any unknown distribution. Our prediction goal is to find the best mapping $F: \mathbb{R}^k \to \mathbb{R}$ that could attain the minimal expected loss $\mathcal{E}(F) := \mathbb{E}\{\mathcal{L}(y, F(\boldsymbol{x}))\}$, where $\mathcal{L}$ denotes the selected Lipschitz continuous loss function and $\boldsymbol{x}$ and y are the test feature vector and test target, respectively. The gradient boosting tree builds a sequence of lower approximations over clusters of binary decision trees. The functional approximation $F^j: \mathbb{R}^k \to \mathbb{R}$ is greedily updated using the last approximation loss as

$$F^j = F^{j-1} + \mu \cdot p^j, \quad (5)$$

where μ is the penalized step size, i.e., learning rate, and p denotes the optimal prediction based on the binary decision tree prediction function, which is shown below.

$$b(\boldsymbol{x}) = \sum_{\ell\in L} \omega \cdot \mathbb{1}_{\{\boldsymbol{x}\epsilon P_\ell\}}, \quad (6)$$

where $\mathbb{1}_{\{\cdot\}}$ denotes the conditional binary operator, L denotes the set of leaves and ω denotes the leaf weight. For each iteration, this function tries to minimize the expected loss in a boosting manner:

$$p^j := \arg\min_b \mathbb{E}\{\mathcal{L}(y, F^{j-1}(\boldsymbol{x}) + b(\boldsymbol{x}))\}. \quad (7)$$

This problem could be effectively solved by typical second-order gradient methods based on the selected loss function. For instance, in our startup/shutdown studies, we choose a simple least-square loss, *i.e.*, $\mathcal{L}(y, y^{pred}) = \frac{1}{2}(y - y^{pred})^2$. Then, by finding its gradient over the prediction $g^j(\boldsymbol{x}, y) = \frac{\partial \mathcal{L}(y, y^{pred})}{\partial y^{pred}}$, following the least-square approximation, we could find the best prediction as

$$p^j := \arg\min_b \mathbb{E}\{(g^j(\boldsymbol{x}, y) - b(\boldsymbol{x}))^2\}. \quad (8)$$

Specifically, for categorical features, we apply the target statistics to use the whole dataset for categorical training [17]. The categorical feature could be replaced with the average label value with feature indication. For example, if x_i^k is categorical, in a random permutation of the dataset, it is replaced by

$$\hat{x}_i^k = \frac{\sum_{c=1}^{p} \mathbb{1}_{\{x_c^k = x_i^k\}} \cdot y_c + \alpha \cdot P}{\sum_{c=1}^{p} \mathbb{1}_{\{x_c^k = x_i^k\}} + \alpha}, \quad (9)$$

where $\alpha > 0$ denotes the step size, $p < n$ is the batch size of the permuted data, and P is the prior value, commonly set as the average value of y [18]. This is an efficient way to reduce the curse of dimensionality raised by the one-hot encoding,

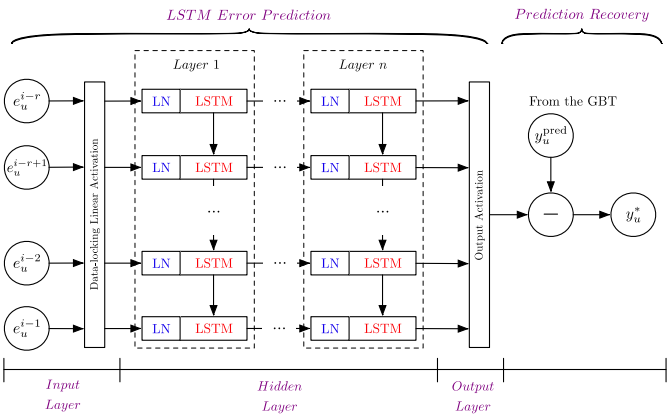


Fig. 2. Online approach for predicting the UDS curves.

whereas it may lead to overfitting since the numeric value $\hat{x}_i^k$ for one batch only contains a part of the categorical information.

While the overfitting issue places a hurdle, during the training of the boosting tree, we enforce an ordered boosting with unbiased gradients [17]. First, the data is batched to two parts $\{A, B\}$ via an order of random permutations. A set of trees is trained for batch A, using the vanilla gradient boosting tree model. As the vanilla gradient boosting tree states, the last approximation loss information is used to update the gradient, making the gradient biased to the previous observations. Hence, another set of trees are trained for batch B. Different batch sizes could be shuffled for the whole dataset. Hence, no direct gradient information between the two batches will be exchanged. Then, batch B model evaluates batch A model in each iteration with scores weighted in the next iteration of training A. This process could make the gradient estimation unbiased in each iteration, which significantly mitigates the overfitting issue caused by the reshaped categorical features and the well-known gradient shift in the boosting tree [19]. For a higher dimension of categorical features, the batch number is also increased.

We could generate high-quality SU/SD curves for LAC using the preprocessed datasets with the Categorical Boosting method. Note that this prediction is purely offline when we apply supervised learning to train on historical data.

B. *Online Approach: Long-short-term Memory Network*

While the offline approach suits the need of the LAC curves, for real-time UDS with finer granularity, the prediction quality of the offline approach needs further improvement. However, the UDS follows a rolling-horizon manner, which means for every five minutes, we could leverage the previously predicted error to enhance the current-interval prediction. This asynchronous error correction could be best tackled by a long-short-term memory (LSTM) network.

The LSTM network is enhanced from the recurrent neural network to mitigate the gradient vanishing problem. Due to its capability of capturing temporal information of data, it has been widely applied for reducing the prediction errors of ML models [20], [21]. We depict the proposed online prediction structure for the UDS SU/SD curves in Fig. 2. It includes an LSTM part and a prediction recovery part. First, we calculate the normalized prediction errors e_u^i of each data entry i for each/ unit u using the gradient boosting tree model as discussed above. The errors are also timeseries, while empty entries are marked with zero flags in the dataset. Then, as shown in the LSTM Error Prediction part of Fig. 2, a stacked LSTM model with multiple hidden LSTM units works for mapping the correlation between the errors using the realized data entries in the history, i.e., $\{e_u^{i-\mathrm{r}}, \ldots, e_u^{i-1}\}$. Each LSTM cell leverages layer normalization (LN) to smooth the activations along the feature direction with whitening. We formally write the training of the LSTM as follows. For each entry i in each layer n, we first compute the forget gate F_i^n, while the output gate O_i^n and the input gate $\mathrm{P_i^n}$ are similarly retained with different weight and bias vectors, i.e., W and b.

$$F_i^n = \sigma(W_f^{n-1,n} h_i^{n-1} + W_f^{n,n} h_{i-1}^n + b_o^n), \tag{10}$$

where σ denotes the sigmoid activation function and h denotes the hidden cell state. Then, based on the activation between the hidden state h and the cell gate C, i.e., $h_i^n = O_i^n \otimes tanh(C_i^n)$, where $\otimes$ denotes the Hadamard product and $tanh$ denotes the hyperbolic tangent activation function, we partially forget the previous cell gate C_{i-1}^n n and update the current C_i^n by a new activated $\tilde{C}_i^n$:

$$\tilde{C}_i^n = tanh(W_c^{n-1,n} h_i^{n-1} + W_c^{n,n} h_{i-1}^n + b_c^n), \tag{11}$$

$$C_i^n = F_i^n \otimes C_{i-1}^n + P_i^n \otimes \tilde{C}_i^n. \tag{12}$$

Then the training proceeds by backpropagating the gradients along each gate path. After the output activation, the predicted timeseries error at the current interval could be used for recovering the actual prediction of the SU/SD value, which will

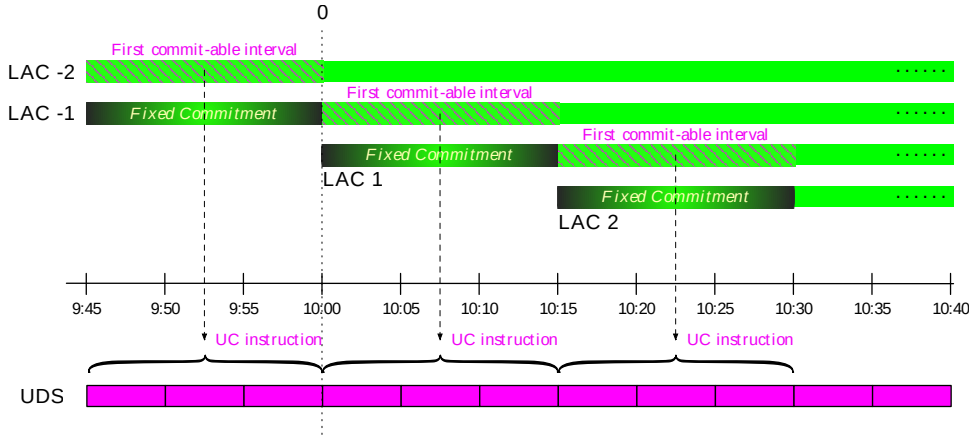


Fig. 3. MISO's LAC and UDS coordination.

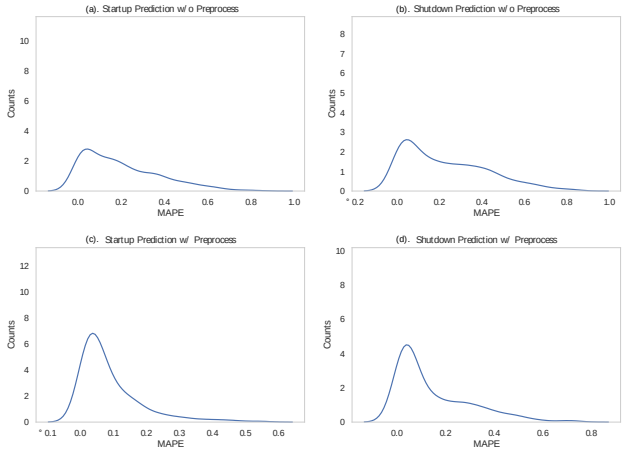


Fig. 4. Prediction performance for each individual unit.

also reveal another error information and add to the next interval's LSTM prediction.

This online prediction process could use the historical data to predict the first few entries, while it recovers higher quality of curve as the time rolls forward because more real-time error data is realized and fed into the model. Hence, it has a great potential to be applied to the UDS rolling-horizon operation.

C. Economic Assessment: Real-time LAC and UDS

After we obtain the SU/SD prediction curves for LAC and UDS as discussed above, we conduct economic assessments on how the predicted curves may benefit ISO/RTO's real time market clearing process. LAC and UDS are the two main real time market clearing processes at MISO. LAC serves as a real-time rolling-horizon UC that commits available fast-responsive units every fifteen minutes, looking ahead three hours, whereas UDS serves as a real-time rolling-horizon ED that follows the commitment decisions from previous commitment processes. Fig. 3 depicts the coordination between LAC and UDS under the MISO practice.

To incorporate the curves, we append the curve information to the generators' active power output variable. Technically, in the LAC model, we add the following constraint to incorporate the predicted SU/SD curves. Note that for LAC, these curves are static and serve as fixed timeseries in the optimization stage.

$$\begin{aligned} p_{g,t} = & \underset{min\{SUT_g, T-t\}}{p_{g,t}^{min+}} + p_g^{min} \cdot i_{g,t} \\ & + \sum_{\substack{i \\ min\{SDT_g, T-t\}}} SUC_i \cdot u_{g,t+i} \\ & + \sum_{i} SDC_i \cdot v_{g,t-i}, \end{aligned} \tag{13}$$

where $p_{g,t}$ denotes the generator's active power output and $p_{g,t}^{min+}$ denotes the scheduled power above the minimum power limit p_g^{min}; $i_{g,t}$, $u_{g,t}$, $v_{g,t}$ denote the variables for commitment, startup, and shutdown, respectively; SUT_g, SDT_g, SUC_g, SDC_g denote the startup time, shutdown time, startup curve value, and shutdown curve value, respectively; T denotes the operation horizon. For a complete detailed description of the MISO's real-time market optimization model including both LAC and UDS, please refer to [1].

Using this revised model, for each LAC instance with a three-hour horizon, it will include the SU/SD curves once the corresponding variables change during the optimization. For cross-instance SU/SDs, we also record the future and past SU/SD periods in the implementation to make the curves consistent in the rolling-horizon operation.

IV. Experiment Results

This section conducts several experiments on the real-world MISO datasets and assesses the predicted curves via benchmarked MISO LAC and UDS cases. We implement the enhanced gradient boosting tree and the LSTM in Python/TensorFlow and carry out the simulated MISO real-time commitment process with plugged-in SU/SD curves using a modified version of EGRET [22]. EGRET is an open-source Python package for electric grid optimization. We customized and benchmarked EGRET functions with the MISO market model for our economic assessments. The MISO test case considers standard UC attributes, including generation capacity limits for 1,200 generators, minimum up/down requirements, ramping limits, power balance, binary startup/shutdown logic, and modeling for transmission power flow and multi-timescale zonal reserves, *etc.*

A. Data Extraction and Feature Selection

We obtained a dataset spanning generator profiles over a period of 4 years for the entire MISO area. Around 800 generators monitored by MISO are included in this study. Note that not all generators in the MISO test case have associated SU/SD data in the 4-year MISO dataset. The data comprises five-minute distributed data with (de-)commitment effective time, state estimation time, state estimation MW values (the prediction target), and other unit characteristics. First, we intuitively select potential feature data for the SU/SD predication model. We perform the feature engineering process for the whole dataset and each unit as discussed in Section II. We take the startup data and test features /for individual unit predictions. We quantitatively evaluate the feature importance by metrics introduced in Section II for each type of unit and then choose features with importance larger than 0.5 in the later machine learning tasks.

Several features for the model were considered, including unit commitment status, unit startup notification time, "off duration" or how many minutes a unit has been offline, and three hourly minimum and maximum power limits, "Pmin" and "Pmax", for emergency, economic, and regulation operation respectively. We observed that all generator types have reasonable dominant features that relate to the state estimation MW. For example, the steam turbines' startup curves are decided mainly by economic Pmin because most steam turbines are operated for economic purposes with large capacities. But the combined-cycle units' curves are more related to the ramp limits as they are most likely to be fast responsive, and the ramping capabilities influence the startup processes.

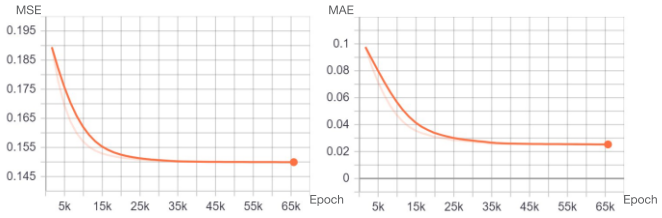
We could also see that the environmental factors, e.g., temperature, and temporal factors, e.g., year, had less impact the startup curves. This may be because non-renewable

TABLE I
RESULT COMPARISON OF LAC & UDS OPERATIONS FOR ONE TYPICAL WEEK

Operation	Production Cost						Penalty Cost					
	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6
Original LAC	0	0	0	0	0	0	0	0	0	0	0	0
Original SLAC	+0.034%				+0.009%		-4.9%				0%	
LAC + Curves	-0.16%	+0.03%	-0.27%	-0.03%	-0.12%	-0.08%	-0.69%	-24.9%	-5.15%	-0.02%	-0.19%	2.02%
SLAC + Curves	-0.24%				-0.12%		-4.6%				+0.5%	

generators are often placed in power plants, and their working environment is mostly static. We excluded these features in the later individual units' training stage. Filtering the features with high feature importance contributes to a higher quality of training, especially for noisy and polluted datasets.

B. Offline Prediction



a). Loss b). Evaluation metric

Fig. 5. Prediction performance for each individual unit.

For the offline training, we first conduct the customized data preprocessing for each individual unit. Then, independent categorical gradient boosting trees, introduced in Section III, are built for predicting each unit's SU/SD curves. We present a comparative study for both SU/SD curves before and after the data preprocessing in Fig. 4. We use the mean absolute percentage error (MAPE) as the evaluation metric for the gradient boosting tree, which is shown below.

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - y_i^{\text{pred}}}{y_i}\right|.$$

Fig. 4 depicts the distribution of units according to its prediction MAPE on the testing set. The preprocessing mentioned here includes the customized individual data cleaning and feature selection discussed in Section II. We also use the same training model for all four cases. We could observe that the preprocessing helps with improving the prediction performance for both the start and shutdown curves, as shown in the estimated probability distribution function of blue lines. Also, note that, within the data cleaning stage, the preprocessing also filters out units with SU/SD time shorter than fifteen minutes, which are not needed for LAC with fifteen-minute resolution. Hence, only a part of the units shown in Fig. 4 (a) and (b) are plotted in Fig. 4. (c) and (d), while the others are excluded from the datasets. This further shows the data cleaning and feature selection processes enhance the overall performance of the SU/SD curves' generation.

C. Online Prediction

We further test the performance of the proposed LSTM network for the online training task. We first perform the offline training and collect all the error information of the training and testing data, which constructs the dataset for the online training task. Then we build the LSTM network as discussed in Section III. B. to update the next-interval prediction. As the probability distribution of the training data is static, we adopt the mean squared error (MSE) as the loss function and the mean absolute error (MAE) as the evaluation metric, which are detailed below.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i^{\text{pred}} - y_i\right)^2,$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y_i^{\text{pred}} - y_i\right|.$$

Fig. 5 depicts the epoch loss and epoch evaluation metric for the startup curve prediction. The bold orange line shows the time-smoothed values, and the faint orange line shows the actual values. The LSTM network has four hidden layers with same structural dimensions. We also implement an intelligent learning rate adjustment scheme based on the training loss, which can automatically reduce the learning rate during the training if it observes weakened loss reduction. The training ends when there is no further loss reduction. As shown in the figure, the evaluation metric MAE could reduce to 2.5% at the end. According to the MAE definition, we could retain the actual prediction errors of the next interval offline prediction within 2.5% in real-time, which will then be used to recover the actual prediction value by substituting the errors with the original offline prediction results. Due to the operation limitation, we skip the online rolling training of the LSTM in UDS but use the existing historical data for the prediction. The overall training time of the LSTM is around 1 hour on an ordinary laptop CPU, but when it comes online after training, it is almost instant to compute the next-interval error.

D. Economic Assessment

After we obtain the SU/SD curves discussed above, we plug in these curves using equation (13) for the LAC and UDS coordinated operation. generators. We studied all units recorded in the MISO state estimation for the SU/SD curve prediction, but not all generators within the operation have associated curves. Some units might have SU/SD times shorter than fifteen minutes, while some units are not frequently committed or de-committed. We choose units with a high prediction performance of SU/SD curves, i.e., MAPE < 10%, in the test case. Whole-day operations with the rolling LAC instances and associated UDS instances are executed for a sample week in this study. The coordination between the LAC and UDS is shown in Fig. 3, while we only incorporate LAC curves for convenience. Apart from the original deterministic LAC, we further test the stochastic LAC (SLAC), which leverages

scenario-based stochastic programming to model the LAC with multiple uncertainty scenarios. Then, we pick two days, *i.e.*, Day 1 and Day 5, to see whether the predicted startup/shutdown curves could synergize with SLAC.

Table I tabulates the original LAC, SLAC optimization results, and their versions using the prototype SU/SD prediction curves, which report their overall production cost and system-wide penalty cost in the whole-day period. The production cost includes the units' SU/SD costs and fuel costs, *etc.*, while the penalty cost consists of the penalty assigned to unserved load, reserve shortfall, and transmission capacity violation, *etc.* We set the original LAC as the base case and report the percentage change of other cases' results from the base case. It is clear from the row of LAC + Curves that using the curves in LAC contributes to overall cost reduction. Only Day 2 has an increase of production cost using the curves, but it achieves the highest penalty reduction among all days. Note that, though the percentage reduction seems tiny, the base value is with the magnitude of tens of millions of dollars. Hence, capturing the SU/SD prediction in the current real-time market clearing process holds the potential to help the operator achieve better market-clearing results and reduce unnecessary penalties of violations.

It is also interesting to find that using the SU/SD prediction curves helps SLAC further reduce the production cost and penalty cost compared with the deterministic LAC. Using SU/SD prediction curves makes committing/de-committing units yield more accurate economic schedules, strengthening SLAC's capability of handling different system scenarios.

V. Concluding Remarks

This paper conducts detailed analyses on predicting units' startup/shutdown curves and investigates their potential contributions to future real-time electricity market operations. Two approaches, *i.e.*, offline prediction and online prediction, are proposed to capture the startup/shutdown uncertainties. The offline approach leverages the historical data to predict static but accurate startup/shutdown curves for the UC module, while the online approach better fits the ED module by updating each interval's prediction based on previous prediction errors. The test results corroborate the efficacy of the proposed approaches and reveal the economic values of adopting the startup/shutdown curves in real-time market clearing processes.

VI. Acknowledgements

The authors would like to thank Jessica Harrison for the support and review of the manuscript.

References

[1] "Business Practices Manual: Energy and Operating Reserve Markets," Tech. Rep. BPM-002-r21, MISO, 2020.

[2] J. M. Kanter and K. Veeramachaneni, "Deep Feature Synthesis: Towards Automating Data Science Endeavors," in 2015 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2015, Paris, France, October 19-21, 2015, pp. 1–10, IEEE, 2015.

[3] C. Cecati, J. Kolbusz, P. R´o˙zycki, P. Siano, and B. M. Wilamowski, "A Novel RBF Training Algorithm for Short-Term Electric Load Forecasting and Comparative Studies," IEEE Transactions on Industrial Electronics, vol. 62, no. 10, pp. 6519–6529, 2015.

[4] N. Tang, S. Mao, Y. Wang, and R. M. Nelms, "Solar Power Generation Forecasting With a LASSO-Based Approach," IEEE Internet of Things Journal, vol. 5, no. 2, pp. 1090–1099, 2018.

[5] Y. Zhang, J. Wang, and B. Chen, "Detecting False Data Injection Attacks in Smart Grids: A Semi-Supervised Deep Learning Approach," IEEE Transactions on Smart Grid, vol. 12, no. 1, pp. 623–634, 2021.

[6] L. Wang, Q. Zhou, and S. Jin, "Physics-guided Deep Learning for Power System State Estimation," Journal of Modern Power Systems and Clean Energy, vol. 8, no. 4, pp. 607–615, 2020.

[7] F. Mohammadi, M. Sahraei-Ardakani, D. Trakas, and N. D. Hatziargyriou, "Machine Learning Assisted Stochastic Unit Commitment during Hurricanes with Predictable Line Outages," IEEE Transactions on Power Systems, pp. 1–1, 2021.

[8] C. Wan, J. Wang, J. Lin, Y. Song, and Z. Y. Dong, "Nonparametric Prediction Intervals of Wind Power via Linear Programming," IEEE Transactions on Power Systems, vol. 33, no. 1, pp. 1074–1076, 2018.

[9] H.-T. Yang, C.-M. Huang, Y.-C. Huang, and Y.-S. Pai, "A WeatherBased Hybrid Method for 1-Day Ahead Hourly Forecasting of PV Power Output," IEEE Transactions on Sustainable Energy, vol. 5, no. 3, pp. 917–926, 2014.

[10] J. R. Andrade and R. J. Bessa, "Improving Renewable Energy Forecasting with a Grid of Numerical Weather Predictions," IEEE Transactions on Sustainable Energy, vol. 8, no. 4, pp. 1571–1580, 2017.

[11] J. Dowell and P. Pinson, "Very-Short-Term Probabilistic Wind Power Forecasts by Sparse Vector Autoregression," IEEE Transactions on Smart Grid, vol. 7, no. 2, pp. 763–770, 2016.

[12] M. Khodayar and J. Wang, "Spatio-Temporal Graph Deep Neural Network for Short-Term Wind Speed Forecasting," IEEE Transactions on Sustainable Energy, vol. 10, no. 2, pp. 670–681, 2019.

[13] X. Lin, Z. J. Hou, Y. Chen, S. Rose, Y. Ma, and F. Pan, "Probabilistic Forecasting of Generators Startups and Shutdowns in the MISO System Based on Random Forest," in 2020 IEEE Power Energy Society General Meeting (PESGM), pp. 1–5, 2020.

[14] P. Cerda, G. Varoquaux, and B. K´egl, "Similarity Encoding for Learning with Dirty Categorical Variables," 2018.

[15] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Reading, Massachusetts: Addison-Wesley, 2009.

[16] O. Chapelle, E. Manavoglu, and R. Rosales, "Simple and Scalable Response Prediction for Display Advertising," ACM Trans. Intell. Syst. Technol., vol. 5, Dec. 2015.

[17] A. V. Dorogush, V. Ershov, and A. Gulin, "CatBoost: Gradient Boosting with Categorical Features Support," CoRR, vol. abs/1810.11363, 2018.

[18] D. Micci-Barreca, "A Preprocessing Scheme for High-Cardinality Categorical Attributes in Classification and Prediction Problems," Association for Computing Machinery, vol. 3, p. 27–32, July 2001.

[19] J. H. Friedman, "Stochastic Gradient Boosting," Comput. Stat. Data Anal., vol. 38, p. 367–378, Feb. 2002.

[20] Y. Liu, J. Duan, and J. Meng, "Difference Attention Based Error Correction LSTM Model for Time Series Prediction," Journal of Physics: Conference Series, vol. 1550, p. 032121, May 2020.

[21] C. Yu, H. Ahn, and J. Seok, "Coordinate-RNN for Error Correction on Numerical Weather Prediction," in 2018 International Conference on Electronics, Information, and Communication (ICEIC), pp. 1–3, 2018.

[22] B. Knueven, J. Ostrowski, and J.-P. Watson, "On Mixed-Integer Programming Formulations for the Unit Commitment Problem," INFORMS Journal on Computing, vol. 32, no. 4, pp. 857–876, 2020.