Reinforced Deterministic and Probabilistic Load Forecasting via *Q*-Learning Dynamic Model Selection

Cong Feng, *Student Member, IEEE*, Mucun Sun, *Student Member, IEEE*, and Jie Zhang, *Senior Member, IEEE*

***Abstract*—Both deterministic and probabilistic load forecasting (DLF and PLF) are of critical importance to reliable and economical power system operations. However, most of the widely used statistical machine learning (ML) models are trained by optimizing the global performance, without considering the local behaviour. This paper develops a two-step short-term load forecasting (STLF) model with Q-learning based dynamic model selection (QMS), which provides reinforced deterministic and probabilistic load forecasts (DLFs and PLFs). First, a deterministic forecasting model pool (DMP) and a probabilistic forecasting model pool (PMP) are built based on 10 state-of-the-art ML DLF models and 4 predictive distribution models. Then, in the first-step of each time stamp, a Q-learning agent selects the locally-best DLF model from the DMP to provide an enhanced DLF. At last, the DLF is input to the best PLF model selected from the PMP by another Q-learning agent to perform PLF in the second-step. Numerical simulations on two-year weather and smart meter data show that the developed STLF-QMS method improves DLF and PLF by 50% and 60%, respectively, compared to the state-of-the-art benchmarks.**

***Index Terms*—Load forecasting, machine learning, reinforcement learning, probabilistic forecasting, dynamic model selection.**

NOMENCLATURE

p, q	The probability and quantile.
$\hat{y}$	A deterministic load forecast.
μ, σ, σ^*, $\hat{\sigma}^*$	The mean, standard deviation, optimal standard deviation, and the pseudo standard deviation of predictive distributions.
$\boldsymbol{A}$, $\boldsymbol{a}$, a_i, a^*	The action space, the action vector, the ith action, and the optimal action.
$\boldsymbol{Q}$, $Q(s,a)$, $\boldsymbol{Q}^*$	The Q-table, Q-table element, and optimal policy.
$\boldsymbol{R}$, $R(s,a)$	The reward matrix and reward function.
$\boldsymbol{S}$, $\boldsymbol{s}$, s_i	The state space, the state vector, and the ith state.
$\boldsymbol{T}_{td}$, $\boldsymbol{T}_{tp}$	The Q-learning training datasets.
$\boldsymbol{T}_{cd}$, $\boldsymbol{T}_{cp}$	The Q-learning processing datasets.
$\mathbf{X}_d$, $\mathbf{X}_{ts}$, $\mathbf{X}_p$	Training datasets for DLF models, SVR surrogate models, and PLF models.
$\mathbf{X}_v$, $\mathbf{X}_E$	The validation and testing datasets.
$\hat{\mathbf{Y}}_{D_i}$, $\hat{\mathbf{Y}}^*{}_D$, $\hat{\mathbf{Y}}^*{}_P$	Vectors of DLFs generated by the ith model, the Q-learing reinforced DLFs, and Q-learing reinforced PLFs.
D_i, P_j	The ith DLF model and the jth PLF model.
Φ_i	The ith DLF algorithm.
Ψ_{P_j}	The SVR surrogate model corresponding to jth predictive distribution.
D_Q, P_Q	The developed Q-learning reinforced DLF model and PLF model.
$\mathrm{M}_{i,j}$, M_{Q^2}	Model with the ith DLF model in the first step and the jth PLF model in the second step, and the developed two-step model with QMS in both steps.
$F(\cdot)$, $F^{-1}(\cdot)$	The CDF function and its corresponding inverse function.
$Q(\cdot)$, $H(\cdot)$	The quantile function and the Heaviside step function.
$L(\cdot)$	The pinball loss function.
$EM(\cdot)$, $RK(\cdot)$	The model evaluation function and ranking funtion.

Manuscript received January 7, 2019; revised June 4, 2019 and August 16, 2019; accepted August 20, 2019. Date of publication August 26, 2019; date of current version February 19, 2020. Paper no. TSG-00033-2019. *(Corresponding author: Jie Zhang.)*

The authors are with the Department of Mechanical Engineering, The University of Texas at Dallas, Richardson, TX 75080 USA (e-mail: cong.feng1@utdallas.edu; mucun.sun@utdallas.edu; jiezhang@utdallas.edu).

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

Digital Object Identifier 10.1109/TSG.2019.2937338

I. INTRODUCTION

ACCURATE short-term load forecasting (STLF), including both deterministic load forecasting (DLF) and probabilistic load forecasting (PLF), plays an important role in reliable and economical power system operations. Typically, DLF can be used in the design of demand response programs, unit commitment, economic dispatch, energy trading, and others [1]. In contrast, PLF provides more valuable uncertainty information than DLF in the circumstance of increasing market competition, aging infrastructure, renewable penetration, and the more active and less predictive electricity market [2]. To better manage the future uncertainty in power systems, PLF has been extensively applied in stochastic unit commitment, probabilistic power flow, and probabilistic transmission planning [3].

The STLF literature mainly focused on DLF in the past decades, which led to a wealth of DLF techniques. DLF methods can be classified into different categories based on forecasting lead time, spatial scales, and method principles.

According to the forecasting method principle, DLF methods can be roughly categorized into statistical methods, machine learning (ML) methods, and deep learning methods. Statistical methods, such as the autoregressive integrated moving average method, are the first generation of DLF, usually built based on only the historical time series [4]. The most popular used DLF methods are ML-based models, e.g., artificial neural network (ANN) and support vector machine (SVR), which are able to integrate external information such as meteorological data. Deep learning methods have been recently applied to STLF, which have shown promising learning abilities [5]. A more comprehensive review of DLF methods can be found in recent review papers [6], [7]. However, one common drawback of most existing data-driven methods is that their parameters are optimized by minimizing the overall accuracy objective function without considering the local performance. For example, Jiang *et al.* [8] optimized SVR models for STLF based on a risk function of all observations with a two-step hybrid parameter searching algorithm. The loss function used to update ANN (including both shallow ANN and deep NN) is usually in terms of mean error, which is calculated based on a batch of samples [5].

PLF has commanded attention only since the most recent decade. According to the source of uncertainties, PLF methods can be roughly categorized into one-step and two-step methods. Specifically, the one-step PLF captures the future uncertainty in the step of generating PLFs by quantile regression (QR) and its variants, such as Gaussian process QR [9] and QRNN [2], [10], or quantile estimation, such as the lower upper bound estimation [11]. On the contrary, the uncertainty of two-step PLF could originate from the diverse input scenarios [12], various input combinations [13], or different point forecasters [14]. A comprehensive review of PLF methods can be found in [15]. Similar to DLF, most of the PLF models are optimized based on the global forecasting performance, without considering the local behavior. For example, Quan *et al.* [11] optimized a NN to generate lower and upper bounds with the overall best coverage width-based objective function. Wang *et al.* [14] determined the optimal weights of single QR methods based on pinball loss (PL) summation. Moreover, QR based PLF methods are in form of least absolute deviations regression, which minimizes a sum of asymmetrically weighted absolute residuals [16].

In addition to the forecasting methods, a variety of methodologies,[1] such as feature selection [17], ensemble forecasting [18], aggregate forecasting [19], and time series decomposition [20], have been developed to further enhance the STLF accuracy, which leads to an even larger collection of options for STLF. Therefore, appropriate and effective model selection for STLF becomes increasingly important, especially with the consensus that a universally best model does not exist [15]. In our previous work, we developed a Q-learning based dynamic model selection (DMS), which significantly improved the DLF accuracy by choosing the best forecasting model at each forecasting time step [21]. This reinforcement learning based DMS work inspires similar research, such as DMS for ensemble learning [22]. Nevertheless, the current STLF model selection is far from maturity, due to: (i) the lack of relevant literature (less than 5 papers returned from a quoted Google Scholar search-"load forecasting model selection"), especially for PLF; (ii) the confusion of model selection with feature selection and hyperparameter optimization, such as in [22], [23], which restricts the scope of available selectable candidates; and (iii) the neglect of heterogeneity between macroscopic superiority and local performance, such as in [17].

With the aim of coping with the two deficiencies in current STLF research, i.e., the over-reliance on global accuracy in STLF and a lack of effective model selection methodology with local awareness, we develop a reinforcement learning based dynamic model selection (QMS) methodology in this paper. The developed QMS relies on the Q-learning agents that are able to select the best future DLF/PLF models at every single forecasting step from a deterministic forecasting model pool (DMP)/probabilistic forecasting model pool (DMP). The main contributions of this paper include: (i) introducing the locally optimized probabilistic forecasting method into PLF domain; (ii) proposing a QMS methodology, which learns from the latest model performance and selects the best future models at each forecasting time step; and (iii) improving the STLF by over 50% with the developed STLF-QMS method, compared to state-of-the-art benchmarks.

The remainder of this paper is organized as follows. The STLF-QMS methodology is described in Section II. Section III briefly summarizes the data source and experiment setups. Numerical simulation results are presented and discussed in Section IV. Section V discusses the flexibility and scalability of the method. Section VI summarizes the conclusions.

II. STLF-QMS Methodology

This section describes the details of the developed STLF, including both DLF and PLF, with Q-learning based dynamic Model Selection (STLF-QMS). The overall framework of STLF-QMS is illustrated in Fig. 1, which consists of three interactive major modules: (i) the deterministic forecasting model pool (DMP), (ii) the probabilistic forecasting model pool (PMP), and (iii) the QMS. The three major modules, along with the detailed description of the STLF-QMS, are orderly described in this section.

A. Short-Term Deterministic Forecasting Model Pool (DMP)

A collection of ML-based short-term deterministic forecasting models constitute the DMP, from which the best model is selected at each time step in the forecasting stage. The DMP consists of ten models with four state-of-the-art ML algorithms [24], i.e., ANN [25], SVR [8], gradient boosting machine (GBM) [26], and random forest (RF) [27], which are further diversified by different training algorithms, kernel functions, or distribution functions. Specifically, three ANN models with standard back-propagation (BP), momentum-enhanced BP, and resilient BP training algorithms are selected based on

[1]Four terms are repeatedly used in this paper, which are *methodology*, *algorithm*, *method*, and *model*. A methodology refers to a general solution framework that can be implemented with different models [15]. Several models can be built based on one method or algorithm.

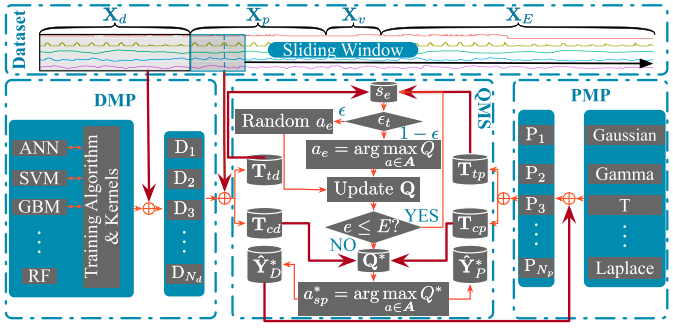


Fig. 1. Framework of the developed STLF with Q-learning based dynamic Model Selection (STLF-QMS).

TABLE I
ML BASED MODELS IN THE DMP

Algorithm	Model	Training algorithm and kernel function
	D_1	Standard back-propagation
ANN	D_2	Momentum-enhanced back-propagation
	D_3	Resilient back-propagation
	D_4	Linear kernel
SVR	D_5	Polynomial kernel
	D_6	Radial basis function kernel
	D_7	Squared loss
GBM	D_8	Laplace loss
	D_9	T-distribution loss
RF	D_{10}	CART aggregation

their fast convergence and satisfactory performance [28]. The most popular kernels in SVR are used, which are linear, polynomial, and radial base function kernels. GBM models with squared, Laplace, and T-distribution loss functions are empirically selected. The last model is an RF model. The details of the models are summarized in Table I. The triplets of DLF models, algorithms, and forecasts in the DMP are denoted as $\{D_i, \Phi_i, \hat{\mathbf{Y}}_{D_i}\}$, where $i = 1, 2, \ldots, N_d$, and $N_d = 10$ is the number of models in the DMP. These models are trained based on a DLF training dataset, $\mathbf{X}_d$, and hyperparameters are tuned using a validation dataset, $\mathbf{X}_v$, as shown in the bottom-left box in Fig. 1. The detailed mathematical descriptions of the four ML algorithms could be found in [29]. The DLF models are selected based on their good performance and popularity. Please note that in this study, we are not trying to identify the existing best-performing DLF models in the literature, since (i) no single algorithm can always be the best in all scenarios, (ii) it's challenging to implement all the methodologies behind the models, such as feature selection, parameter optimization, transfer learning, etc., (iii) the best model also has a risk to generate large local errors. Therefore, we focus on reinforcing the performance of the most widely-used, easily-implemented, and standard machine learing models.

B. Short-Term Probabilistic Forecasting Model Pool (PMP)

The PMP contains a total of four two-step parametric PLF models, which take a DLF ($\hat{y}_t$) as input and output its cumulative distribution function (CDF) at every time step [30]. Four distributions, parameterized by means (μ) and standard deviations (σ), are selected due to their ability of quantifying uncertainty in time series forecasting, which are normal,

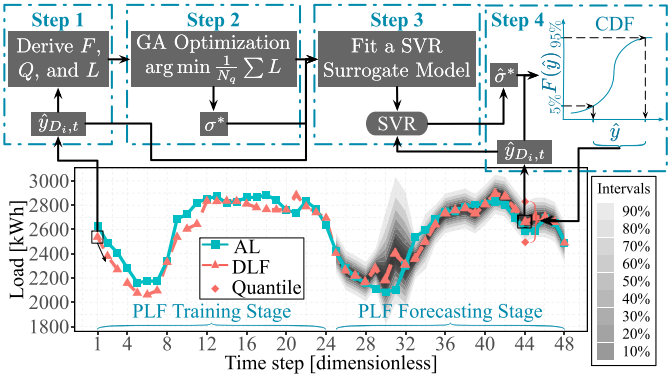


Fig. 2. Flowchart of the training and forecasting procedure of a PLF model. AL: actual load.

Gamma, Laplace, and noncentral-t distributions. The DLF uncertainty is captured by minimizing PL of the CDF-derived quantiles at each forecasting step. The procedure of PLF training and forecasting is illustrated in Fig. 2 and described as follows:

- ***Step 1:*** Parameterizing uncertainty of the DLF, $\hat{y}$, in terms of μ and σ, where μ is assumed to be a DLF. Then, the quantile, q, and its corresponding pinball loss, L_q, are derived and expressed by q and σ:

$$\begin{aligned} F_{D_i,P_j,t}(\hat{y}_t|\mu_t, \sigma_t) &= F_{D_i,P_j,t}(\mu_{D_i,P_j,t}, \sigma_{D_i,P_j,t}) \\ &= F_{D_i,P_j,t}(\hat{y}_{D_i,t}, \sigma_{D_i,P_j,t}) \\ &= F_{D_i,P_j,t}(\sigma_{D_i,P_j,t}) \end{aligned} \tag{1}$$

$$Q_{D_i,P_j,t}(p) = F^{-1}_{D_i,P_j,t}(p) = F^{-1}_{D_i,P_j,t}(p, \sigma_{D_i,P_j,t}) \tag{2}$$

$$\begin{aligned} L_{D_i,P_j,q,t}(q, \sigma_{D_i,P_j,t}) = &-q\{Q_{D_i,P_j,t}(q) - y_t\} \\ &- H(y_t - Q_{D_i,P_j,t}(q)) \end{aligned} \tag{3}$$

  where D_i, P_j indicate the DLF and PLF models, respectively, $i = 1, \ldots, N_d$ and $j = 1, \ldots, N_p$ ($N_p = 4$); t is a time index; $F(\cdot)$ and $F^{-1}(\cdot)$ are a CDF function and its corresponding inverse function; $Q(\cdot)$ is the quantile function; p and q are probability and a quantile, respectively; $H(\cdot)$ is the Heaviside step function.
- ***Step 2:*** Optimizing the DLF uncertainty indicator, σ (the only unknown parameter), at each forecasting time step by minimizing the average pinball loss of all quantiles with the genetic algorithm (GA) [31]:

$$\begin{aligned} \sigma^*_{D_i,P_j,t} = \underset{\sigma_{D_i,P_j,t}}{\arg\min} \ & \frac{1}{N_q}\sum_{q=1}^{N_q} L_{D_i,P_j,q,t}(q, \sigma_{D_i,P_j,t}) \\ s.t. \quad & \zeta_1 < \sigma_{D_i,P_j,q,t} < \zeta_2 \end{aligned} \tag{4}$$

  where σ^* is the optimized standard deviation; $N_q = 99$ is the number of quantiles; ζ_1 and ζ_2 are the lower and upper bounds of σ, which are 0.01 and 100, respectively.
- ***Step 3:*** An SVR surrogate model is constructed to fit the observation and σ^* set $\{\mathbf{X}_{ts,t}, \sigma^*_t\}_{t=1}^{N_{ts}}$ in the training stage, which is used to generate unknown pseudo standard deviations, $\hat{\sigma}^*$, in the forecasting stage. N_{ts} is the training data length of SVR. $\mathbf{X}_{ts,t} = (\sigma^*_{t-N_l}, \ldots, \sigma^*_{t-1}, \hat{y}_{t-N_l}, \ldots, \hat{y}_t)$ is

input to the SVR surrogate model, where N_l is the lagging length. To fit an SVR model, a radial basis function is applied to transform the $(2N_l + 1)$-dimensional input data into a higher-dimensional linearly-separable feature space [32]:

$$K\left(\mathbf{X}'_{ts,t}, \mathbf{X}_{ts,t}\right) = \exp\left(-\frac{1}{2\delta^2}\left\|\mathbf{X}'_{ts,t} - \mathbf{X}_{ts,t}\right\|^2\right) \quad (5)$$

where δ is a free parameter. Then, a linear regression is applied, given by:

$$\Psi_{P_j}\left(\mathbf{X}'_{ts,t}\right) = \left\langle \omega^T, K\left(\mathbf{X}'_{ts,t}, \mathbf{X}_{ts,t}\right)\right\rangle + b \quad (6)$$

where Ψ_{P_j} is the SVR surrogate model corresponding to jth predictive distribution. ω and b are obtained by minimizing the following objective function [32]:

$$\begin{aligned} \{\omega, b\} = \arg\min \ & \frac{1}{2}\omega^T\omega + C\sum\nolimits_{t=1}^{N_{ts}}\left(\xi_t + \xi_t^*\right) \\ s.t. \quad & \xi_t, \xi_t^* \geq 0 \\ & \sigma_t^* - \left(\left\langle \omega^T, K\left(\mathbf{X}'_{ts,t}, \mathbf{X}_{ts,t}\right)\right\rangle + b\right) \leq \varepsilon + \xi_t \\ & \left(\left\langle \omega^T, K\left(\mathbf{X}'_{ts,t}, \mathbf{X}_{ts,t}\right)\right\rangle + b\right) - \sigma_t^* \geq \varepsilon + \xi_t^* \end{aligned} \quad (7)$$

where ξ_t and ξ_t^* are slack variables; ε is the insenitive parameter and C is the penalty weight.

- ***Step 4:*** In this last step (forecasting stage), a DLF, $\hat{y}_{D_i,t}$, is first generated by a DLF model. Then, $\hat{y}_{D_i,t}$, along with other observations, are fed into the SVR surrogate model to get the pseudo standard deviation, $\hat{\sigma}^*_{D_i,P_j,t} = \Psi_{D_i,P_j}(\mathbf{X}_{ts,t})$. At last, the CDF and quantiles of the DLF at that time are derived by Eqs. (1) and (2), respectively, which are visualized as predictive intervals (PIs) in Fig. 2.

C. Q-Learning Based Dynamic Model Selection (QMS)

Once DLFs and PLFs are generated by models in the DMP and PMP, the best DLF and PLF models are selected respectively by two reinforcement learning agents at each forecasting time step. Reinforcement learning is a typical ML algorithm that models an agent interacting with its environment. In this paper, Q-learning, a model-free adaptive dynamic programming algorithm, is adopted to learn the optimal policy of finding the best DLF and PLF models at every forecasting time step.

In order to train a Q-learning agent, a mathematical framework of QMS is first defined in a Markov Decision Process (MDP). In general, a Q-learning agent takes sequential *actions* at a series of *states* based on a state-action value matrix, *Q-table*, until reaching an ultimate goal [33]. The actions are evaluated by a scalar *reward* feedback returned from the environment, which is used to update the Q-table. In this research, the state space, $\boldsymbol{S}$, is composed of the possible forecasting models (DLF or PLF) at the current time:

$$\boldsymbol{S} = \{\boldsymbol{s}\} = \left\{s_1, s_2, \ldots, s_{N_s}\right\} \quad (8)$$

where s_i means the current forecasting model is D_i (or P_i if it is used in QMS for PLF, denoted as PLF-QMS); N_s is the number of states, which equals the number of selectable models, N_d or N_p. Similarly, the action space, $\boldsymbol{A}$, is composed of the selectable forecasting models for the next time step:

$$\boldsymbol{A} = \{\boldsymbol{a}\} = \left\{a_1, a_2, \ldots, a_{N_a}\right\} \quad (9)$$

where a_j means taking the action of switching from the current forecasting model to D_j (or P_j if it is used in PLF-QMS) at the next forecasting time step; N_a is the number of action options, which is identical to N_s in this paper.

To successfully solve an MDP using Q-learning, the most important step is to maintain a reward matrix, $\boldsymbol{R}$, by a proper reward function, $R(s, a)$. Two reward strategies are considered in this paper, which are based on (i) the improvement of next-state forecast over the current-state forecast, and (ii) the performance ranking improvement of the next-state model over the current-state model (the ranking of the best model is 1). The corresponding reward functions of the two strategies are:

$$R_t\left(s_i, a_j\right) = EM\left(D_{i,t}\right) - EM\left(D_{j,t+1}\right) \quad (10a)$$

$$R_t\left(s_i, a_j\right) = RK\left(D_{i,t}\right) - RK\left(D_{j,t+1}\right) \quad (10b)$$

where $D_{i,t}$ can be replaced by $P_{i,t}$ in PLF-QMS. Two evaluation metrics, absolute percentage error and average pinball loss, are respectively adopted according to the QMS objective (for DLF or PLF):

$$EM(\cdot) = \begin{cases} EM_b\left(D_{i,t}\right) = \dfrac{\left|\hat{y}_{i,t} - y_i\right|}{y_i} & (11a) \\ EM_{pl}\left(P_{i,t}\right) = \dfrac{1}{N_q}\sum\limits_{q=1}^{N_q} L_{P_i,q,t}(\cdot) & (11b) \end{cases}$$

The convergence test of the Q-learning with two reward strategies is performed and the learning curves are shown in Fig. 3. It's observed that Q-learning with the reward function in Eq. (10a) fails to converge, which is shown as red lines. This is because the magnitude of forecasting evaluation metrics does not only depend on forecasting models but is also changing with time. Taking the action of switching from a worse model to the best model might still receive a negative reward (due to the deterioration of forecasting evaluation metrics). In contrast, Q-learning with the reward function in Eq. (10b), shown as the blue lines in Fig. 3, converges successfully, since the model ranking improvement is only related

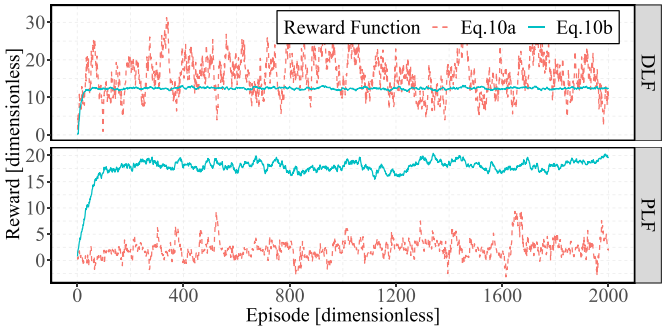


Fig. 3. The Q-learning learning curves (in terms of reward curves) of two reward strategies in Eqs. (10a) and (10b). Q-learning for PLF model selection converges slower than that for DLF model selection, since the PLF model ranking is based on pinball loss, which is more complex than the DLF model ranking criterion, i.e., absolute percentage error.

Algorithm 1 Q-Learning Based Dynamic Model Selection (QMS)

Require:
Number of steps, N_{sp}, in a QMS procedure
Model pool dimension N_m, which is either N_d or N_p
Q-learning training dataset $\boldsymbol{T}_{td}$ or $\boldsymbol{T}_{tp} \in \mathbb{R}^{N_q \times N_m}$
QMS process dataset $\boldsymbol{T}_{cd}$ or $\boldsymbol{T}_{cp} \in \mathbb{R}^{N_{sp} \times N_m}$
Learning rate α, discount factor γ, number of episodes N_e

Ensure: Select the best model from N_m models at each step in $\boldsymbol{T}_{cd}$ or $\boldsymbol{T}_{cp}$

1: Initialize $\boldsymbol{Q} = \overrightarrow{\boldsymbol{0}}_{N_m \times N_m}$, $\epsilon = 1$
2: **for** e = 1 **to** N_e **do**
3: With the probability of ϵ select a random action a_e, otherwise select $a_e = \arg\max_{a \in \boldsymbol{A}} Q_e(s_e, a)$
4: Calculate $\boldsymbol{R}$ by Eq. 10b
5: Update $\boldsymbol{Q}$ by

$$Q_{e+1}(s_e, a_e) = (1-\alpha)Q_e(s_e, a_e) + \alpha[R_e(s_e, a_e) + \gamma \max_{a \in \boldsymbol{A}} Q_e(s_{e+1}, a)] \quad (12)$$

6: $\epsilon \leftarrow \epsilon - \frac{1}{N_e}$
7: **end for**
8: **for** sp = 1 **to** N_{sp} **do**
9: Take action $a^*_{sp} = \arg\max_{a \in \boldsymbol{A}} Q^*(s_{sp}, a)$
10: **end for**

to the model, which avoids the time series effects [19] of the load. Therefore, in this paper, we design reward function as the model performance ranking improvement, which ensures the effective and efficient convergence of Q-learning.

With state, action, and reward defined, the QMS is realized by training Q-learning agents using the Q-learning training datasets $\boldsymbol{T}_{td}$/$\boldsymbol{T}_{tp}$, which are applied to the QMS process datasets $\boldsymbol{T}_{cd}$/$\boldsymbol{T}_{cp}$, as detailed in Algorithm 1 and illustrated in the bottom middle box of Fig. 1. The critical component of determining (steps 1-7 in Algorithm 1) and applying (steps 8-11 in Algorithm 1) the QMS policy is the Q-table, $\boldsymbol{Q}$, which contains triplets of s, a, and $Q(s, a)$. As shown in Algorithm 1, Q values are initialized to be zero and updated repeatedly by Eq. (12) based on the action reward in the current state and the maximum reward in the next state, where α is the learning rate that controls the aggressiveness of learning, γ is a discount factor that weights the future reward. The balance of exploitation and exploration in Q-learning is maintained by adopting a decaying ϵ_t-greedy method [34]. The Q-learning agent with the decaying ϵ_t-greedy method takes completely random actions at the beginning, while reducing the randomness with a decaying ϵ during the learning process. The Q-learning algorithm will eventually converge to the optimal policy, $\boldsymbol{Q}^*$, after N_e iterations, which is applied to find the optimal actions, a^*, in the QMS process.

D. The STLF With QMS

The developed STLF-QMS method has two steps, i.e., DLF-QMS and PLF-QMS, which is denoted as $M_{Q^2} = D_Q + P_Q$. As shown in Fig. 1, the dataset is divided into four parts: (i) DLF training data, $\mathbf{X}_d$, which is used to train N_d DLF models; (ii) PLF training data, $\mathbf{X}_p$, which is used to train N_p PLF models; (iii) validation data, $\mathbf{X}_v$, which is used to tune DLF hyperparameters, PLF (ζ_1, ζ_2, δ, ε, and C) and Q-learning (α, γ, N_q, and N_{sp}) parameters; and (iv) testing data, $\mathbf{X}_E$, which is used to validate the effectiveness of the developed DLF and PLF with QMS. In addition, a sliding window with a length of $(N_q + N_{sp})$ is used to select and partition data for QMS. Specifically, data segments of episodes are generated from the first N_q samples and used to train two Q-learning agents (one for DLF model selection and one for PLF model selection). Then, the optimal policy is adopted to make model selection decisions for the next N_{sp} steps. The stride of the sliding window is identical to the length. Note that the features of input to different models are different.

The workflow of the developed STLF-QMS model is illustrated in Fig. 1. In the training stage, DLF models are first trained with $\mathbf{X}_d$, thereafter providing DLFs, $\hat{\mathbf{Y}}_D$. Then, QMS is dynamically trained and applied to select the best DLF model at every single step. The selected best DLF models generate DLFs, $\hat{\mathbf{Y}}^*{}_D$, which are also the input to PLF candidate models for training. Multiple PLF models are trained and serve as the input to PLF-QMS, which is the last step in the training stage. In the testing (actual forecasting) stage, DLF and PLF models are adaptively selected by two Q-learning agents at every forecasting step, which generate final reinforced DLF and PLF, $\hat{\mathbf{Y}}^*{}_D$ and $\hat{\mathbf{Y}}^*{}_P$, respectively.

III. Experiment Setup

A. Data Description

In this paper, hourly load data of the University of Texas at Dallas (UTD)[2] is used for 1-hour-ahead (1HA) STLF [35]. The reasons to research 1HA STLF with university campus load are twofold: (i) 1HA STLF is used in various power system operations, which is also scalable to longer-term STLF; (ii) demand-side STLF is more challenging than upper-level STLF in power systems [36], and (iii) large electricity consumers, such as universities, are more critical in demand-side management. In addition to campus load, hourly weather information is retrieved from the National Solar Radiation Database (NSRDB).[3] The weather features in NSRDB dataset include air temperature, relative humidity, air pressure, wind speed, wind direction, direct normal irradiance, global horizontal irradiance, and diffuse horizontal irradiance. Calendar features, i.e., hour of the day, day of the week, and month of the year, are also extracted and included in the case studies to capture the calendar patterns [19]. The data from January 12$^{\text{th}}$ 2015 to December 31$^{\text{st}}$ 2015 is used for testing.

Both load and weather data spans from January 1$^{\text{st}}$ 2014 to December 31$^{\text{st}}$ 2015. The data ranging from January 1$^{\text{st}}$ 2014 to November 30$^{\text{th}}$ 2014 is used to train DLF models, while the data from December 1$^{\text{st}}$ 2014 to December 15$^{\text{th}}$ 2014 is used to train PLF models. The data from December 16$^{\text{th}}$ 2014 to January 8$^{\text{th}}$ 2015 is used to tune forecasting model

[2]https://doi.org/10.21227/jdw5-z996
[3]https://nsrdb.nrel.gov

TABLE II
DLF MODEL HYPERPARAMETERS

Model	Hyperparameters
D_1	lr=0.01, max_epoch=1,000
D_2	lr=0.01, max_epoch=1,000, momentum=0.9
D_3	lr=0.01, max_epoch=1,000, min_delta=1×10^6, max_delta=50
D_4	C_d=0.1, ε_d=0.001
D_5	C_d=0.1, ε_d=0.001, δ_d=0.1, degree=3
D_6	C_d=0.1, ε_d=0.001, δ_d=0.1
D_7	lr=0.01, ntrees=1,000, max_depth=20, bag_frac=0.5
D_8	lr=0.01, ntrees=1,000, max_depth=20, bag_frac=0.5
D_9	lr=0.01, ntrees=1,000, max_depth=20, bag_frac=0.5, DF=4
D_{10}	ntrees=1,000, mtry=5

hyperparameters, SVR surrogate model parameters, and the Q-learning parameters.

B. Parameters & Hyperparameters

Specifically, SVR surrogate parameters are $\delta = 0.001$, $\varepsilon = 0.001$ and $C = 1$; the Q-learning parameters are $\alpha = 0.1$ and $\gamma = 0.8$, which ensures the learning speed (thus, $N_e = 100$) and also respects the future reward. The moving window parameters in Q-learning are set as: $N_{sp} = 4$, $N_q = 72$. The DLF model hyperparameters are empirically determined based on the validation dataset and are listed in Table II, including the learning rate (lr) and the maximum number of epochs (max_epoch) in D_1–D_3; the momentum (momentum) in D_2, the minimum update value (min_delta), and the maximum update value (max_delta) in D_3; the penalty weight (C_d) and insentive parameter (ε_d) in D_4–D_6; the free parameter (δ_d) in D_5 and D_6; the degree of the polynomial (degree) in D_5; the number of boosting iterations (ntrees), maximum tree depth (max_depth), learning rate (lr), out-of-bag fraction (bag_frac) in D_7–D_9; the degree of freedom (DF) in D_9; and the number of trees (ntrees) and the number of variables randomly sampled as candidates at each split (mtry) in D_{10}.

C. Comparisons & Implementation

The developed M_{Q^2} model generates both DLFs and PLFs, therefore, is compared to DLF and PLF benchmarks. To validate the effectiveness of the M_{Q^2} model, two sets of competitive models are selected: (i) candidate models described in Sections II-A and II-B, and (ii) the widely-used ensemble learning models that leverages multiple candiate models. Specifically, the D_Q method is compared to $D_i (i = 1, \ldots, N_d)$ and the Machine Learning-based Multi-Model forecasting framework (M3) models [29] with linear regression and the three best ensemble algorithms in the second layer (denoted as B_1–B_4). Moreover, M_{Q^2} is compared to $M_{i,j}$, where $i = \{Q, 1, 2, \ldots, N_d\}$ and $j = \{Q, 1, 2, \ldots, N_p\}$, but i, j can not equal Q at the same time (denoted as $\mathbf{M}_{-Q^2}$). QR and QR averaging (QRA) models are adopted to compare with the M_{Q^2} PLF model. Please note the training data for D_i and ensemble models is 3:1 in ensemble benchmarks.

The case studies are conducted on a laptop with an Intel Core i7-4870HQ CPU running at 2.6 GHz and with 16.0 GB RAM. The DLF models, PLF models, and GA optimization are implemented with caret, rmutil, quantreg, e1071, and GA packages, in R version 3.5.1.

D. Evaluation Criteria

The DLF models are evaluated by three popular used forecasting errors, i.e., normalized mean absolute error (*nMAE*), mean absolute percentage error (*MAPE*), and normalized root mean square error (*nRMSE*) [29]. To assess the PLF effectiveness, reliability, sharpness, and comprehensive performance of PLF PI and quantiles are evaluated. The reliability and sharpness of PLF are respectively quantified by PI coverage probability (PICP) and interval score (IS) of every PI, while the comprehensive judgement that considers both reliability and sharpness is made by the normalized average of all the quantiles' pinball loss (nPL) [14]. The smaller nPL and IS indicate a better PLF model, while PICP should be close to its corresponding PI nominal confidence (PINC) [37].

In addition to performance evaluation metrics, the improvement of the developed method over the benchmarks is also of importance. The improvement based on one of the above metrics is expressed as:

$$Imp_k\left(\frac{M_{Q^2}}{M_{i,j}}\right) = \frac{EV_k(M_{i,j}) - EV_k(M_{Q^2})}{EV_k(M_{i,j})} \times 100\% \quad (13)$$

where $EV(\cdot)$ is an evaluation function; $k \in \{a, p, r, l\}$, which means the improvement is calculated based on *nMAE*, *MAPE*, *nRMSE*, and nPL, respectively.

IV. RESULTS

A. DLF Performance

1) Q-Learning Convergence: The effectiveness of the DLF-QMS is evaluated using the testing dataset with 354 days in 2015. Based on the sliding window parameters, a Q-learning agent is trained every four time steps to make DMS. Therefore, there are totally 2,124 Q-learning agents built to select proper DLF models for the 8,496 time steps in 2015. Figure 4 shows the statistics of Q-learning agent learning curves, which indicates the fast and successful convergence of Q-learning agents. Specifically, Q-learning agents learn extremely fast from interactions with the environment in the first 40 episodes. After the first 40 episodes, even though the exploration probability is still high ($\epsilon = 0.6$ when $e = 40$), Q-learning agents learn slowly and tend to converge. Thus, Q-learning agents converge effectively and efficiently in the DLF-QMS.

2) DLF-QMS Effectiveness: To verify the effectiveness of the DLF-QMS, the rankings of each model at every time step of one year are counted and statistically shown as a violin plot in Fig. 5. It is observed from the figure that forecasting models perform distinctively at different time steps, where every model could become the best or the worst at a certain time step. Each model also shows unique characteristics. For example, the three ANN models (D_1, D_2, D_3) rank 8th, 9th, 10th (the worst three) and 1st, 2nd (the best two) more times than other rankings. An SVR model (D_6) almost has the same chance for each ranking. It's important to note that no single model (D_1-D_{10}) dominates others in the DLF. The effectiveness of the DLF-QMS is evident by comparing the violin of

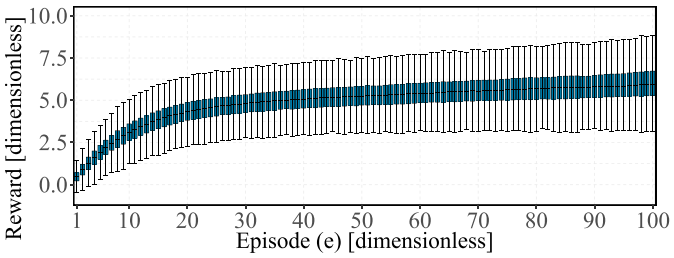


Fig. 4. Learning curve statistics of 2,124 Q-learning agents for DLF-QMS. The reward of each episode is the summation of Q-values in Q-table. Lines in the boxes are the medians. The interquartile range box represents the middle 50% of the rewards. The upper and lower bounds are maximum and minimum values of the rewards, respectively.

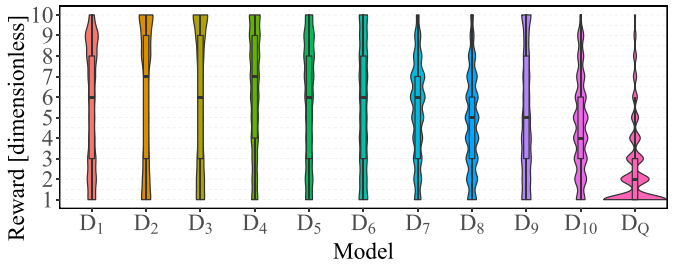


Fig. 5. Violin plot of DLF model rankings. The changing width of a violin indicates the distribution of rankings of a forecasting model, while the boxplot inside a violin shows similar statistical information as Fig. 4.

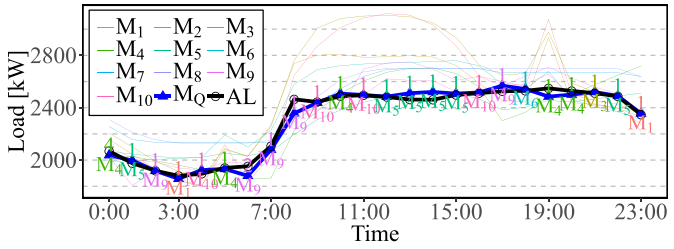


Fig. 6. Forecasting and actual load (AL) time series of one day; values above forecasting points are rankings of the selected models; symbols below forecasting points are names of the selected models; the annotation font color and the line color of the same model are identical.

D_Q with violins of other models. It is found that Q-learning agents select top 3 models at each forecasting step with a 75% chance and they tend to select a better model, since D_Q model has an upward violin.

Figure 6 shows the actual and forecasting time series of one day. Specifically, the thin lines represent 10 DLF models in the DMP, the thick blue line represents the developed D_Q in M_{Q^2}, and the thick black line represents the actual load (AL) time series. In general, most selected models rank 1st except for the models at 0am and 6am. However, the Q-learning agent intends to select the 4th model at 0am since it will receive a large reward ($R(s_4, a_5) = 3$) by switching from D_4 (ranking 4th) at 0am to D_5 (ranking 1st) at 1am. It is also logic for the Q-learning agent to select D_9 at 6am because it values the future reward of this selection.

3) Overall DLF Accuracy: The overall performance of the developed M_{Q^2} method (also denoted as D_Q) and its benchmarks are evaluated and compared with 6 metrics, which are listed in Table III. It is first found that different ML models

TABLE III
FORECASTING ERRORS [%] AND IMPROVEMENTS [%] OF THE DEVELOPED D_Q AND BENCHMARK MODELS

Model	Forecasting Error			Forecasting Improvement		
	$nMAE$	$MAPE$	$nRMSE$	Imp_a	Imp_p	Imp_r
D_1	6.74	11.53	9.05	57.85	57.27	54.39
D_2	7.41	12.71	9.85	61.67	61.26	58.08
D_3	7.81	13.50	10.66	**63.64**	**63.52**	**61.28**
D_4	7.20	11.58	9.12	60.58	57.47	54.73
D_5	6.58	10.77	8.21	56.83	54.28	49.7
D_6	6.35	10.22	8.21	55.31	51.81	49.68
D_7	5.89	9.97	7.41	51.76	50.6	44.26
D_8	5.47	9.37	7.12	48.06	47.42	41.97
D_9	6.20	10.12	8.07	54.18	51.31	48.85
D_{10}	**5.27**	**8.74**	**6.68**	46.14	43.67	38.19
B_1	6.98	11.15	8.97	59.30	55.83	53.96
B_2	6.03	9.99	7.86	52.89	50.71	47.44
B_3	5.70	9.33	7.45	50.22	47.22	44.54
B_4	5.47	9.29	7.32	48.11	47.00	43.62
D_Q	**2.84**	**4.93**	**4.13**	***57.96***	***55.38***	***52.82***

Note: **Bold values** indicate the best benchmark models or the most improvements of the developed model over candidate models, **bold green values** indicate the developed M_{Q^2} model, and ***italic values*** indicate the average improvements.

and models with the same ML algorithm but different training or kernel functions have distinctive overall performance. For example, ensemble ML models (i.e., D_7-D_{10}) are generally better than ANN and SVR models (i.e., D_1-D_6), while SVR with RBF kernel produces more accurate DLFs than those with linear and polynomial kernels. Ensemble learning models reduce the risk of unsatisfactory models but cannot beat the best candidate model. Most importantly, Imp_a, Imp_p, and Imp_r indicate the improvement of the developed M_{Q^2} method over benchmarks, from which significant enhancements are observed. The average DLF improvements are more than 50%, which confirm that the local performance awareness of the DLF-QMS is effective.

B. PLF Performance

1) Q-Learning Convergence & PLF-QMS Effectiveness: Similar to DLF, effectiveness of the developed M_{Q^2} PLF method is first confirmed by verifying Q-learning convergence and QMS success. Figure 7 shows that the convergence pattern of Q-learning agents for PLF-QMS is similar to that of Q-learning agents for DLF-QMS. This is due to the stable rewarding strategy we designed in Eqs. (10b)-(11a). The successful convergence also leads Q-learning agents effectively perform the PLF-QMS, which are illustrated by violin plots in Fig. 8. The enhancement of PLF-QMS is validated by $\mathbf{M}_{i,Q}$ (i.e., D_i+ P_Q) ranking improvements over benchmarks ($\mathbf{M}_{i,-Q}$).[4] It's found that $\mathbf{M}_{i,Q}$ has more 1st rankings than others.

2) Overall PLF Accuracy: The normalized average pinball losses (nPLs) of the developed M_{Q^2} model and benchmarks $\mathbf{M}_{-Q^2}$ are summarized in Table IV, based on which the improvements of the M_{Q^2} model over $\mathbf{M}_{-Q^2}$ models are shown in Fig. 9a. The developed M_{Q^2} model generates better PLF

[4]The subscript $-Q$ means all models, excluding the model with QMS. For example, $\mathbf{M}_{i,-Q}$ is a PLF model set with any model in the fist-step and any model but the PLF-QMS model in the second-step.

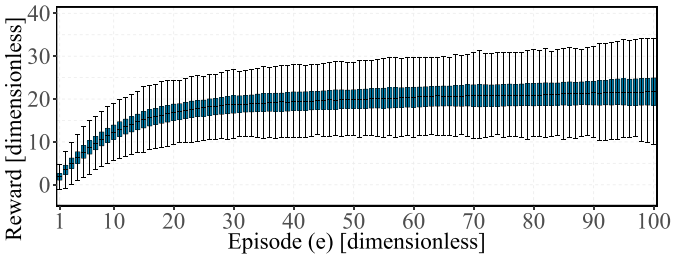


Fig. 7. Learning curve statistics of 2,124 Q-learning agents for PLF-QMS. The boxes have the same meaning as those in Fig. 4.

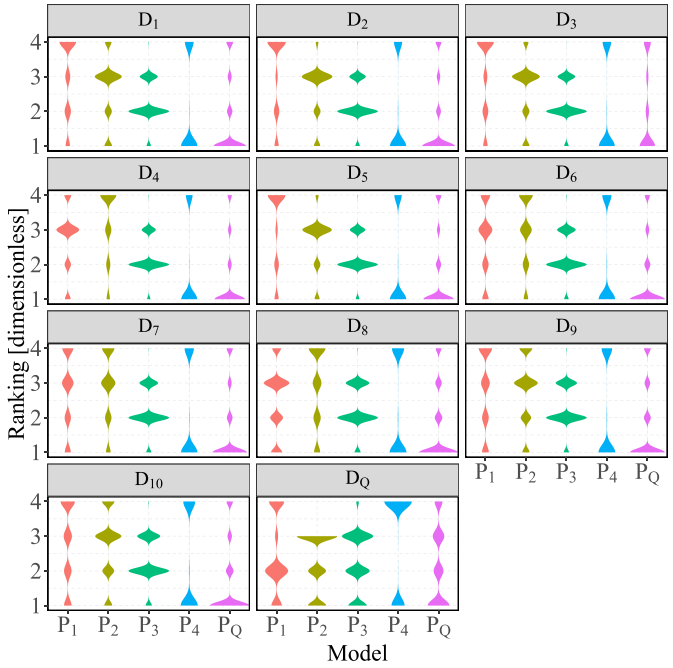


Fig. 8. Violin plot of forecasting model rankings. The facet title of each subplot is the DLF model used in the two-step PLF, while the abscissa axis indicates the PLF model used in the two-step PLF.

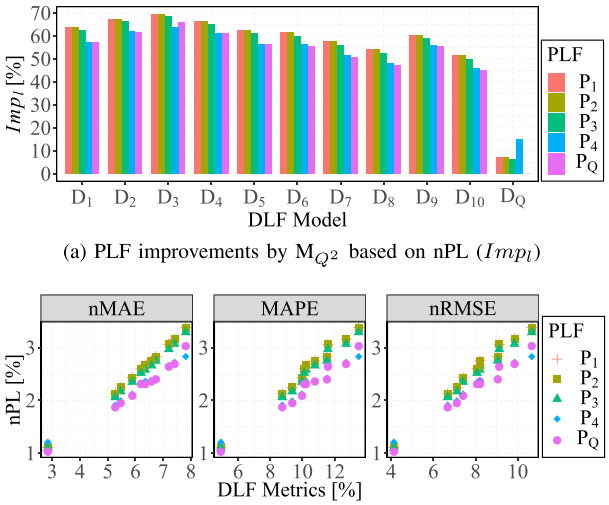
(nPL = 1.03%) than any of the competing models, since the Imp_l values are positive and the average Imp_l is 54.21%. Specifically, the average Imp_l of the M_{Q^2} model over $\mathbf{M}_{i,-Q}$ models and $\mathbf{M}_{i,Q}$ models are 58.92% and 8.92%, respectively. The difference of the improvement magnitude is because of the significant impact of the first-step DLF model performance on the second-step PLF model in the two-step PLF, which is shown in Fig. 9b. Figure 10 shows the quantiles' PICP and IS of the $\mathbf{M}_{i,Q}$ and $\mathbf{M}_{Q,j}$ models, from which we found the superior reliability and satisfactory sharpness of the developed M_{Q^2} model. To this end, it is concluded that QMS effectively and immensely enhances both the DLF and PLF accuracy.

3) PLF Time Series: PLF provides future uncertainty in terms of quantile intervals, which are valuable to power system operators for decision-makings. Five models, i.e., baseline model, best $\mathbf{M}_{-Q,-Q}$, $\mathbf{M}_{i,Q}$, $\mathbf{M}_{Q,j}$ models, and the developed M_{Q^2} model, are selected to detail the PLF quantile time series, which are QR, $M_{10,4}$, $M_{10,Q}$, $M_{Q,3}$, and M_{Q^2}. Figure 11 visualizes the AL, DLF, and PLF time series, from which the one-step QR is the worst due to its inaccurate DLFs and redundant PLF uncertainty. Compared to $M_{10,4}$ and $M_{10,Q}$, the

TABLE IV
NORMALIZED AVERAGE PINBALL LOSS (NPL) [%] OF THE DEVELOPED M_{Q^2} MODEL AND BENCHMARKS $\mathbf{M}_{-Q^2}$

M_{ij}	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}	D_Q
P_1	2.83	3.16	3.38	3.06	2.75	2.67	2.43	2.24	2.60	2.13	***1.11***
P_2	2.83	3.16	3.38	3.07	2.75	2.67	2.43	2.25	2.60	2.13	***1.11***
P_3	2.75	3.07	3.29	2.97	2.66	2.58	2.35	2.17	2.52	2.06	***1.10***
P_4	2.41	2.72	2.83	2.66	2.37	2.36	2.12	1.99	2.33	*1.91*	***1.21***
P_Q	**2.40**	**2.69**	**3.03**	**2.64**	**2.36**	**2.31**	**2.09**	**1.95**	**2.31**	**1.87**	***1.03***

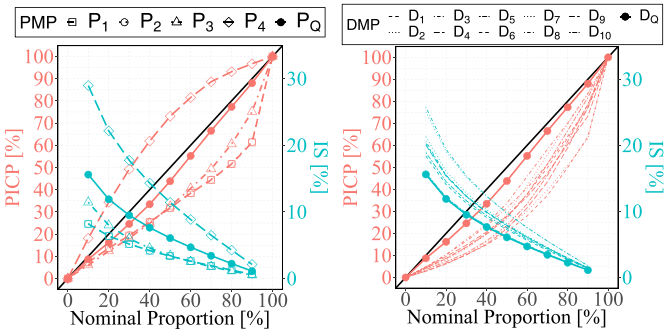
Note: **Bold values** indicate $\mathbf{M}_{i,Q}$ models, ***bold italic values*** indicate $\mathbf{M}_{Q,j}$ models, **bold italic green values** indicate the developed M_{Q^2} model, and the ***italic values*** indicate the best model without any QMS. The nPLs of QR and QRA are 5.28% and 3.65%, respectively.



(a) PLF improvements by M_{Q^2} based on nPL (Imp_l)

(b) Relationship between DLF error metrics and PLF nPL

Fig. 9. PLF performance analysis.



(a) Reliability and sharpness curves of D_Q with different PLF

(b) Reliability and sharpness curves of P_Q with different DLF

Fig. 10. PICP and IS of quantiles. The black diagonal line is the PINC curve [37].

first-step D_Q of $M_{Q,3}$ and M_{Q^2} reduces the deviations between DLFs and AL. Moreover, it's observed that the second-step P_Q adjusts the uncertain magnitude based on DLF bias. For example, $M_{10,Q}$ decreases interval widths when DLF is accurate (e.g., time step 1-12 and 25-36) compared to $M_{10,4}$, which has the same DLF model but without PLF-QMS. On the contrary, $M_{10,Q}$ adds more uncertainty to compensate the DLF deviations from time step 38 to 48, compared to $M_{Q,3}$. Therefore,

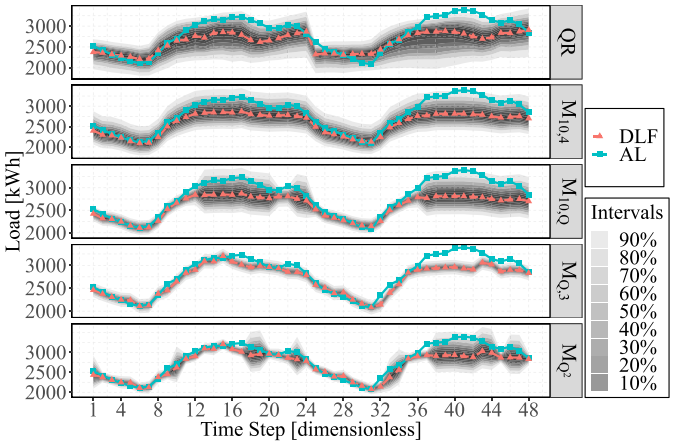


Fig. 11. AL, DLF, and PLF time series of 5 selected methods.

it's concluded that M_{Q^2} provides the best PLF because it utilizes enhanced DLF and dynamically adjusts future uncertainty based on DLF bias.

V. Discussion

The developed reinforcement learning enhanced forecasting model has been verified to be effective for 1HA STLF. However, the method is sufficiently general, flexible, and scalable to be applied to different forecasting tasks.

A. Forecasting Target

Formed in similar regression problems, the same forecasting method is usually applicable to different forecasting targets, e.g., load, wind, and solar. For example, NN was used for both load and wind forecasting in [38]. Sáez *et al.* [39] generated load, wind, and solar forecasts for a microgrid with fuzzy prediction interval models. The developed reinforcement learning enhanced forecasting model is flexible with the forecasting target, since (i) the forecasting models in model pools have been used in wind [40], [41] and solar forecasting [42]–[44], and (ii) the QMS does not rely on the optimal overall accuracy but the local diverse performance of forecasting models, which is satisfied in the forecasting with different targets.

B. Time Horizon

Forecasts with different time horizons are valuable to various time-scale power system operations and markets. This paper focuses on the 1HA forecasting due to its flexibility and scalability. There are three ways to extend the short-term forecasting into longer-term forecasting, namely, the recursive approach, the parallel approach, and the combination of the two approaches [45]–[47]. Although different approaches show superiorities under different circumstances, the parallel approach is more popular in multi-step ahead load forecasting, as shown in [48], [49]. Relying on the DLF and PLF models, the developed method can be applied to longer-term forecasting by all the three approaches.

C. Spatial Scale

The customer-level load data was used for case studies, which is more challenging than the higher-level forecasting in the power system hierarchy [50]. Therefore, it is expected that the developed method will perform better for higher-level load forecasting (e.g., transformers, feeders, substations, etc). To validate this point and add more credibility to the paper, the proposed method has applied to the data of GEFCom2014 load forecasting task 1. The *nMAE*, *MAPE*, and *nRMSE* of the developed D_Q are 1.88%, 3.82%, and 2.42%, while corresponding forecasting errors of the best DLF benchmark are 3.09%, 6.26%, and 5.38%, respectively. For the PLF, nPL of the developed M_{Q^2} is 0.68%, while nPL of the best PLF benchmark withouth Q-learning is 1.19%.

TABLE V
Computational Costs of the Developed Method

Stage	Process	Computational time (min)
Training	DLF model training	43.74
	PLF model training	8.02
	Q agents training	2.16×10^{-1}/4 steps
Forecasting	DLF forecasting	2.28×10^{-5}/step
	PLF forecasting	9.51×10^{-6}/step
	QMS	4.58×10^{-5}/step

One challenge for large spatial-scale distribution-level or customer-level load forecasting is the computational cost. Table V summarizes the computational time of the six processes in the case studies. The forecasting models took 51.76 min to train, which are usually updated less-frequently in real applications. Two Q agents were trained every four steps, which took averagely 0.22 min. It took around 4.69 ms to generate a DLF and a PLF with the developed method. In general, the proposed method is extendable to the system-level data with fine spatial granularity and large coverage, especially with the development of distributed computing frameworks, such as Apache Hadoop and Spark.

VI. Conclusion

This paper developed a reinforcement learning enhanced two-step short-term load forecasting (STLF) model, which provides both deterministic and probabilistic load forecasts (DLFs and PLFs). Ten state-of-the-art machine learning DLF models constituted a deterministic forecasting model pool (DMP) and four genetic algorithm optimized predictive distribution surrogate models composed a probabilistic forecasting model pool (PMP). At each time step, a Q-learning agent selected the best DLF model from the DMP to provide DLF in the first-step, which was input to the best PLF model selected from the PMP by another Q-learning agent to perform PLF in the second-step. Numerical simulations on two-year weather and smart meter data showed that:

(1) Both Q-learning in DLF and PLF model selections (DLF-QMS and PLF-QMS) effectively selected the optimal DLF and PLF models at each forecasting step;
(2) The QMS enhanced LF by improving DLF accuracy and dynamically adjusting PLF uncertainty;
(3) The developed STLF-QMS model reduced DLF errors by over **50%** and improved PLF accuracy by nearly **60%**.

Future work will focus on comparing model selection capabilities of different reinforcement learning, such as State-Action-Reward-State-Action and deep Q-learning.

REFERENCES

[1] T. Saksornchai, W.-J. Lee, K. Methaprayoon, J. R. Liao, and R. J. Ross, "Improve the unit commitment scheduling by using the neural-network-based short-term load forecasting," *IEEE Trans. Ind. Appl.*, vol. 41, no. 1, pp. 169–179, Jan./Feb. 2005.

[2] M. Rafiei, T. Niknam, J. Aghaei, M. Shafie-Khah, and J. P. S. Catalão, "Probabilistic load forecasting using an improved wavelet neural network trained by generalized extreme learning machine," *IEEE Trans. Smart Grid*, vol. 9, no. 6, pp. 6961–6971, Nov. 2018.

[3] M. Rouhani, M. Mohammadi, and A. Kargarian, "Parzen window density estimator-based probabilistic power flow with correlated uncertainties," *IEEE Trans. Sustain. Energy*, vol. 7, no. 3, pp. 1170–1181, Jul. 2016.

[4] R. Weron, *Modeling and Forecasting Electricity Loads and Prices: A Statistical Approach*, vol. 403. Chichester, U.K.: Wiley, 2007.

[5] H. Shi, M. Xu, and R. Li, "Deep learning for household load forecasting—A novel pooling deep RNN," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 5271–5280, Sep. 2018.

[6] M. Q. Raza and A. Khosravi, "A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings," *Renew. Sustain. Energy Rev.*, vol. 50, pp. 1352–1372, Oct. 2015.

[7] B. Yildiz, J. I. Bilbao, and A. B. Sproul, "A review and analysis of regression and machine learning models on commercial building electricity load forecasting," *Renew. Sustain. Energy Rev.*, vol. 73, pp. 1104–1122, Jun. 2017.

[8] H. Jiang, Y. Zhang, E. Muljadi, J. J. Zhang, and D. W. Gao, "A short-term and high-resolution distribution system load forecasting approach using support vector regression with hybrid parameters optimization," *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3341–3350, Jul. 2018.

[9] Y. He, Q. Xu, J. Wan, and S. Yang, "Short-term power load probability density forecasting based on quantile regression neural network and triangle kernel function," *Energy*, vol. 114, pp. 498–512, Nov. 2016.

[10] W. Zhang, H. Quan, and D. Srinivasan, "An improved quantile regression neural network for probabilistic load forecasting," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 4425–4434, Jul. 2019.

[11] H. Quan, D. Srinivasan, and A. Khosravi, "Uncertainty handling using neural network-based prediction intervals for electrical load forecasting," *Energy*, vol. 73, pp. 916–925, Aug. 2014.

[12] J. Xie and T. Hong, "Temperature scenario generation for probabilistic load forecasting," *IEEE Trans. Smart Grid*, vol. 9, no. 3, pp. 1680–1687, May 2018.

[13] B. Liu, J. Nowotarski, T. Hong, and R. Weron, "Probabilistic load forecasting via quantile regression averaging on sister forecasts," *IEEE Trans. Smart Grid*, vol. 8, no. 2, pp. 730–737, Mar. 2017.

[14] Y. Wang, N. Zhang, Y. Tan, T. Hong, D. S. Kirschen, and C. Kang, "Combining probabilistic load forecasts," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 3664–3674, Jul. 2019.

[15] T. Hong, P. Pinson, S. Fan, H. Zareipour, A. Troccoli, and R. J. Hyndman, "Probabilistic energy forecasting: Global energy forecasting competition 2014 and beyond," *Int. J. Forecasting*, vol. 32, no. 3, pp. 896–913, 2016.

[16] Y. Yang, S. Li, W. Li, and M. Qu, "Power load probability density forecasting using Gaussian process quantile regression," *Appl. Energy*, vol. 213, pp. 499–509, Mar. 2018.

[17] J. Xie and T. Hong, "Variable selection methods for probabilistic load forecasting: Empirical evidence from seven states of the United States," *IEEE Trans. Smart Grid*, vol. 9, no. 6, pp. 6039–6046, Nov. 2018.

[18] S. Li, L. Goel, and P. Wang, "An ensemble approach for short-term load forecasting by extreme learning machine," *Appl. Energy*, vol. 170, pp. 22–29, May 2016.

[19] C. Feng and J. Zhang, "Short-term load forecasting with different aggregation strategies," in *Proc. Int. Design Eng. Tech. Conf. Comput. Inf. Eng. Conf. (ASME)*, 2018, pp. 1–10.

[20] B. A. Høverstad, A. Tidemann, H. Langseth, and P. Öztürk, "Short-term load forecasting with seasonal decomposition using evolution for parameter tuning," *IEEE Trans. Smart Grid*, vol. 6, no. 4, pp. 1904–1913, Jul. 2015.

[21] C. Feng and J. Zhang, "Reinforcement learning based dynamic model selection for short-term load forecasting," in *Proc. IEEE Power Energy Soc. Innov. Smart Grid Technol. Conf. (ISGT)*, 2019, pp. 1–5.

[22] M. Ma, B. Jin, S. Luo, S. Guo, and H. Huang, "A novel dynamic integration approach for multiple load forecasts based on Q-learning algorithm," *Int. Trans. Elect. Energy Syst.*, Jul. 2019, Art. no. e12146.

[23] Z. Hu, Y. Bao, and T. Xiong, "Comprehensive learning particle swarm optimization based memetic algorithm for model selection in short-term load forecasting using support vector regression," *Appl. Soft Comput.*, vol. 25, pp. 15–25, Dec. 2014.

[24] T. Hong, P. Pinson, and S. Fan, "Global energy forecasting competition 2012," *Int. J. Forecasting*, vol. 30, no. 2, pp. 357–363, 2014.

[25] T. K. Chau, S. S. Yu, T. Fernando, H. H.-C. Iu, and M. Small, "A load-forecasting-based adaptive parameter optimization strategy of STATCOM using ANNs for enhancement of LFOD in power systems," *IEEE Trans. Ind. Informat.*, vol. 14, no. 6, pp. 2463–2472, Jun. 2018.

[26] A. J. Landgraf, "An ensemble approach to GEFCom2017 probabilistic load forecasting," *Int. J. Forecast.*, to be published.

[27] C. Li, Y. Tao, W. Ao, S. Yang, and Y. Bai, "Improving forecasting accuracy of daily enterprise electricity consumption using a random forest based on ensemble empirical mode decomposition," *Energy*, vol. 165, pp. 1220–1227, Dec. 2018.

[28] C. N. Bergmeir and J. M. B. Sánchez, "Neural networks in R using the Stuttgart neural network simulator: RSNNS," *J. Stat. Softw.*, vol. 46, no. 7, pp. 1–26, 2012.

[29] C. Feng, M. Cui, B.-M. Hodge, and J. Zhang, "A data-driven multi-model methodology with deep feature selection for short-term wind forecasting," *Appl. Energy*, vol. 190, pp. 1245–1257, Mar. 2017.

[30] M. Sun, C. Feng, E. K. Chartan, B.-M. Hodge, and J. Zhang, "A two-step short-term probabilistic wind forecasting methodology based on predictive distribution optimization," *Appl. Energy*, vol. 238, pp. 1497–1505, Mar. 2019.

[31] T. Back, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford, U.K.: Oxford Univ. Press, 1996.

[32] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[33] J. Yan, H. He, X. Zhong, and Y. Tang, "Q-learning-based vulnerability analysis of smart grid against sequential topology attacks," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 1, pp. 200–210, Jan. 2017.

[34] V. Mnih *et al.*, "Playing Atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[35] J. Zhang and C. Feng, *Short-Term Load Forecasting Data With Hierarchical Advanced Metering Infrastructure and Weather Features*, IEEE, Piscataway, NJ, USA, 2019. doi: 10.21227/jdw5-z996.

[36] S. B. Taieb, J. Yu, M. N. Barreto, and R. Rajagopal, "Regularization in hierarchical time series forecasting with application to electricity smart meter data," in *Proc. AAAI*, 2017, pp. 4474–4480.

[37] C. Wan, Z. Xu, P. Pinson, Z. Y. Dong, and K. P. Wong, "Probabilistic forecasting of wind power generation using extreme learning machine," *IEEE Trans. Power Syst.*, vol. 29, no. 3, pp. 1033–1044, May 2014.

[38] H. Quan, D. Srinivasan, and A. Khosravi, "Short-term load and wind power forecasting using neural network-based prediction intervals," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 2, pp. 303–315, Feb. 2014.

[39] D. Sáez, F. Ávila, D. Olivares, C. Cañizares, and L. Marín, "Fuzzy prediction interval models for forecasting renewable resources and loads in microgrids," *IEEE Trans. Smart Grid*, vol. 6, no. 2, pp. 548–556, Mar. 2015.

[40] C. Feng, M. Sun, M. Cui, E. K. Chartan, B.-M. Hodge, and J. Zhang, "Characterizing forecastability of wind sites in the United States," *Renew. Energy*, vol. 133, pp. 1352–1365, Aug. 2019.

[41] C. Feng, E. K. Chartan, B.-M. Hodge, and J. Zhang, "Characterizing time series data diversity for wind forecasting," in *Proc. IEEE/ACM 4th Int. Conf. Big Data Comput. Appl. Technol. (BDCAT)*, 2017, pp. 113–119.

[42] C. Feng, M. Cui, B.-M. Hodge, S. Lu, H. Hamann, and J. Zhang, "Unsupervised clustering-based short-term solar forecasting," *IEEE Trans. Sustain. Energy*, to be published.

[43] C. Feng *et al.*, "Short-term global horizontal irradiance forecasting based on sky imaging and pattern recognition," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, 2017, pp. 1–5.

[44] C. Feng, D. Yang, B.-M. Hodge, and J. Zhang, "OpenSolar: Promoting the openness and accessibility of diverse public solar datasets," *Solar Energy*, vol. 188, pp. 1369–1379, Aug. 2019.

[45] S. B. Taieb, G. Bontempi, A. F. Atiya, and A. Sorjamaa, "A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition," *Expert Syst. Appl.*, vol. 39, no. 8, pp. 7067–7083, 2012.

[46] M. G. De Giorgi, M. Malvoni, and P. M. Congedo, "Comparison of strategies for multi-step ahead photovoltaic power forecasting models based on hybrid group method of data handling networks and least square support vector machine," *Energy*, vol. 107, pp. 360–373, Jul. 2016.

[47] D. Fay, J. V. Ringwood, M. Condon, and M. Kelly, "24-h electrical load data—A sequential or partitioned time series?" *Neurocomputing*, vol. 55, nos. 3–4, pp. 469–498, 2003.

[48] S. Fan and L. Chen, "Short-term load forecasting based on an adaptive hybrid method," *IEEE Trans. Power Syst.*, vol. 21, no. 1, pp. 392–401, Feb. 2006.

[49] S. Fan and R. J. Hyndman, "Short-term load forecasting based on a semi-parametric additive model," *IEEE Trans. Power Syst.*, vol. 27, no. 1, pp. 134–141, Feb. 2011.

[50] R. Sevlian and R. Rajagopal, "A scaling law for short term load forecasting on varying levels of aggregation," *Int. J. Elect. Power Energy Syst.*, vol. 98, pp. 350–361, Jun. 2018.