A Meta-learning based Distribution System Load Forecasting Model Selection Framework

Yiyan Li, Member, IEEE, Si Zhang, Student Member, IEEE, Rongxing Hu, Student Member, IEEE, and Ning Lu, Senior Member, IEEE

Abstract—This paper presents a meta-learning based, automatic distribution system load forecasting model selection framework. The framework includes the following processes: feature extraction, candidate model labeling, offline training, and online model recommendation. Using user load forecasting needs as input features, multiple meta-learners are used to rank the available load forecast models based on their forecasting accuracy. Then, a scoring-voting mechanism weights recommendations from each meta-learner to make the final recommendations. Heterogeneous load forecasting tasks with different temporal and technical requirements at different load aggregation levels are set up to train, validate, and test the performance of the proposed framework. Simulation results demonstrate that the performance of the meta-learning based approach is satisfactory in both seen and unseen forecasting tasks.

Index Terms—distribution system, load forecasting, machine learning, meta-learning, model selection.

I. INTRODUCTION

The needs for load forecasting (LF) have increased drastically at all levels in power distribution systems accompanied with the increasing penetration of the distributed generation resource (DER). In distribution systems, LF tasks have a wide range of temporal and technical requirements and are at different load aggregation levels, making the tasks heterogeneous in nature. Although many LF models have been developed in the literature, very few attempts were made towards the development of an automated, credible and robust forecasting framework. The framework includes the following processes: feature extraction, candidate model labeling, offline training, and online model recommendation. Using user load forecasting needs as input features, multiple meta-learners are used to rank the available load forecast models based on their forecasting accuracy. Then, a scoring-voting mechanism weights recommendations from each meta-learner to make the final recommendations. Heterogeneous load forecasting tasks with different temporal and technical requirements at different load aggregation levels are set up to train, validate, and test the performance of the proposed framework. Simulation results demonstrate that the performance of the meta-learning based approach is satisfactory in both seen and unseen forecasting tasks.

Traditionally, the Knowledge-based expert system (KES) approach is used for selecting forecasting models [1]-[4]. The main disadvantage of the KES approach is inflexibility. Whenever new models are introduced or new forecasting scenarios are considered, a manual update of the system rules is required, making the maintenance costs high. Moreover, KES cannot be used for unseen LF tasks. Thus, the KES approach is inadequate for selecting an LF model in an active distribution network (ADN), where LF tasks are heterogeneous in terms of scale, input data characteristics, and LF requirements.

In recent years, meta-learning [5][6], generally interpreted as ‘learning to learn’, is introduced to provide model recommendations for different machine learning tasks. In [10], Matijas et al. prepare 7 candidate models to deal with 65 forecasting tasks where statistical features are created to quantify these tasks and classical classifiers are applied to construct the mapping from task features to optimal models. In [11], Arjmand et al. test a similar system with 6 candidate models and 18 features on 30 forecasting tasks generated from zonal data of Ontario, Canada, where ReliefF is used to assist feature selection and improve the accuracy of model recommendation. In [12], Wang et al. consider both rule-based and meta-learning methods to support the forecasting model selection for univariate time series, where self-organization Map (SOM) is introduced to create and visualize forecasting tasks. Also focusing on univariate time series forecasting, Talagala et al. proposed a similar work in [13] using a larger feature set on more candidate models and tests conducted on monthly, quarterly and yearly time series are considered as different forecasting tasks. In [14], Lemke and Gabrys combine several forecasting methods based on the ranking results provided by the meta-learning system to enhance the forecasting stability. In [15], Heng et al. introduce the framework of meta-learning to wind power forecasting and demonstrate that the meta-learning based approach outperforms individual forecasting models.

There are three main technical issues in the aforementioned approaches. First, most of the aforementioned approaches are based on so called Rice’s structure introduced in [16], which represents only a particular case of meta-learning. The essential question of “how to define the LF model selection as a meta-learning problem?” is not well addressed. Second, significant ambiguity exists in meta-learner selection. In Rice’s structure, the key part of the model selection system is to let the meta-learner establish an effective mapping from the task features to the optimal model recommendation. Existing approaches often select a classical classification algorithm to serve as the meta-learner. Because different classifiers target different expertise areas, a weighting mechanism is needed to combine their assessments for making the final recommendation. Third, without a rigorous LF task set up criterion, the performance of the meta-learning based approach cannot be properly quantified. For example, only tens of toy cases are used in one study whereas hundreds of similar forecasting tasks are used in another. Consequently, the model selection accuracy can range...
anywhere from 20% to 90% depending on which test cases are used for quantifying the performance of a trained meta-learner.

To overcome those technical issues, we propose a meta-learning based, automatic distribution system load forecasting model selection framework. The framework includes the following processes: feature extraction, candidate model labeling, offline training, and online model recommendation. Using user load forecasting needs as input features, multiple meta-learners are used to rank the available load forecast models based on their forecasting accuracy. Then, a scoring-voting mechanism weights recommendations from each meta-learner to make the final recommendations.

The contributions of this paper are threefold. First, we propose a generalized meta-learning approach with rigorous mathematical formulation for solving power system load forecasting problems. Second, we introduce a scoring-voting mechanism for combining the strength of multiple meta-learners, which significantly increase the recommendation accuracy. Third, we developed a procedure for training test case selection and setup to improve the training efficiency.

II. PROBLEM FORMULATION

The framework (see Fig. 1) consists of two layers: a base-learning layer and a meta-learning layer. In the base-learning layer, J learning tasks are created. In the J pairs of data samples, \((X_j, y_j)\), \(X_j\) is the input time series data of an LF model with a dimension of \(N_j \times M\), \(y_j\) is the actual load with a dimension of \(N_j \times 1\), and \(j \in \{1, \ldots, J\}\). To conduct each LF task, we divide \(X_j\) into \(X_{j\text{train}}\) and \(X_{j\text{test}}\), and \(y_j\) into \(y_{j\text{train}}\) and \(y_{j\text{test}}\) so that \(N_j - K_j\) samples are used for training and \(K_j\) samples are used for testing.

For an LF task, there are \(I_B\) LF models serving as candidate LF algorithms, as shown in the base-learning layer schematic in Fig. 1. Each LF model will be trained using \(X_{j\text{train}}\) and \(y_{j\text{train}}\) and tested using data set \(X_{j\text{test}}, y_{j\text{test}}\). The model accuracy is calculated using the root-mean-square error (RMSE) between \(y_{j\text{test}}\) and \(y_{j\text{test}}^\ast\). The LF model with the smallest RMSE will be selected as the model to be used for this training task.

Thus, after we complete the training and testing for all \(J\) LF tasks in the base-learning layer, the best performed LF model for each LF task, \(\Phi\), is considered known and labeled. The input feature matrix of the meta-learning layer, \(\mathbf{F}\), has a dimension of \(J \times D\). The input features of each LF task consist of two parts: input data statistics, \(\mathbf{F}(j, 1: R)\) and technical requirements of the LF task \(\mathbf{F}(j, R + 1: D)\).

As shown in the meta-learning layer schematic in Fig. 1, the meta-data obtained from the base-learning layer, \(\mathbf{F}, \Phi\), is divided into a training meta-data set, \((\mathbf{F}_{\text{train}}, \Phi_{\text{train}})\) and a testing meta-data set \((\mathbf{F}_{\text{test}}, \Phi_{\text{test}})\). There are \(I_M\) meta-learners used, so \(I_M\) sets of recommendations, \((\Phi_{\text{IM}})\), will be obtained. Then, a voting engine that weights \(\Phi_{\text{w}}\) by predicted accuracy of each meta-learner will be used to determine the optimal \(\Phi\). In the following subsections, we will introduce the problem formulation of the base-learning layer LF model selection process and the meta-learning layer LF model recommendation mechanisms and illustrate the online application procedure.

A. Base-learning Layer Problem formulation

In the machine learning domain, power system LF problems belong to supervised machine learning. The set of training data \(X_j\) for the \(i\)th LF task can be represented as

\[
x_{j\text{train}} = \begin{bmatrix} x_{1j}^1 & x_{1j}^2 & \cdots & x_{1j}^M \\ x_{2j}^1 & x_{2j}^2 & \cdots & x_{2j}^M \\ \vdots & \vdots & \ddots & \vdots \\ x_{nj}^1 & x_{nj}^2 & \cdots & x_{nj}^M \end{bmatrix}_{(N_j \times M)}
\]

where \(x_{nj}^m\) represents the \((n, j, m)\)th attributes of the \(n\)th sample. Using \(X_{j\text{train}}\) as the forecasted load generated by LF model \(i_B\) for the \(i\)th LF task, the base-learning layer problem can be formulated as

\[
\hat{y}_{j\text{train}} = f_{i_B}(X_{j\text{train}})
\]

\[
\theta_i = \arg \min_{\theta} \mathcal{L}_{\text{base}}(\hat{y}_{j\text{train}}, y_{j\text{train}})
\]

where \(\mathcal{L}_{\text{base}}\) is the loss function calculated as distance between the actual load \(y_{j\text{train}}\) and the predicted load \(\hat{y}_{j\text{train}}\), and \(\theta_i\) is the optimal parameters for LF model \(i_B\). Once \(\theta_i\) is obtained, the forecasting accuracy is measured by the RMSE errors on the testing data set, so we have

\[
\hat{y}_{j\text{test}} = f_{i_B}(X_{j\text{test}})
\]

\[
\mathcal{L}(i_B, i) = \left\| \hat{y}_{j\text{test}} - y_{j\text{test}} \right\|_2
\]

The LF model with the highest accuracy among all \(I_B\) LF models is selected as the recommended LF model for the \(i\)th LF task and its index is stored in \(\Phi(i)\).

B. Meta-learning Layer Problem Formulation

By summarizing cross-task knowledge into meta-knowledge, a meta-learner can learn ‘how to learn tasks’ from known tasks in order to improve its performance in new tasks [17]. Meta-knowledge can be in different forms, for example, selecting algorithms or optimizers to solve different tasks [18] and finding initialization parameters for different machine learning models [6].

In this paper, meta-learning is used to find the best LF model for a LF task. The problem is formulated as

\[
\Phi_{\text{train}} = g_w^\ast(f_{\text{train}})
\]

\[
w^\ast = \arg \min_w \mathcal{L}_{\text{meta}}(\Phi_{\text{train}}, \Phi_{\text{train}})
\]

where \(g_w\) represents the meta-learner with parameter \(w\), \(\mathcal{L}_{\text{meta}}\) is the loss function measuring the distance between the actual best LF model \(\Phi_{\text{train}}\) and the recommended LF model by meta-learner \(g_w^\ast, \Phi_{\text{train}}\).

Once the optimal parameters \(w^\ast\) is determined, the performance of the meta-learner will be tested on the testing set. The recommendation accuracy, \(\eta_{IM}^\ast\), is calculated as

\[
\Phi_{\text{test}} = g_w^\ast(f_{\text{test}})
\]

\[
\eta_{IM}^\ast = \frac{1}{J-1} \sum_{j=K_M+1}^{J} \| \Phi_{\text{test}}(j) - \Phi_{\text{test}}(\Phi^\ast) \|^2
\]

Because multiple meta-learners are used to cover the diversity in LF tasks, recommendations from different meta-learners, \(\Phi_{\text{IM}}, i_M \in [1, \ldots, I_M]\), need to be weighted through a scoring-voting mechanism in order to obtain the final model.
The accuracy of the final model recommendation, $\eta$, is calculated as

$$\eta = \frac{1}{J-K_M}\sum_{j=K_M+1}^{J} \phi_{test}(j)$$

(10)

C. Online application and framework extension

After the training is finished, the framework can be applied online for recommending one or a few LF models for new LF tasks. First, the feature set of the new LF task, $F_{new}$, is calculated. Then, recommendations from all meta-learners, $g_{w^*}(F_{new}), i \in [1, \ldots, I_M]$, will be sent to the voting engine to obtain the final LF model recommendation, $\phi_{new}$. The online application involves only forward calculation so it is very computationally efficient. The main advantage of the meta-learning based approach is its extendibility because a user can readily incorporate new task samples, LF models, meta-features and meta-learners into the existing framework, making it scalable and low maintenance.

III. IMPLEMENTATION SETUP

This section introduces the implementation setup of the proposed meta-learning LF model selection framework.

A. LF task setup

To learn how to select the best LF models for unseen LF tasks, it is critical for the meta-learners to be trained and tested on a large amount of heterogeneous distribution system LF tasks. In this paper, we consider that LF Tasks differ from one another in five aspects: data granularity, data length, forecasting horizon, exogenous factors, and load aggregation level. Thus, a building-level day-ahead LF task with 1-year hourly load and temperature data sets as inputs can be described by a red dashed line in the 5-dimensional radar chart in Fig. 2. By randomly select values of the variable representing the five LF task features, a wide range of heterogeneous LF tasks can be created.

![Fig. 1. Flowchart of the proposed meta-learning based LF model selection framework.](image-url)

![Fig. 2. Five main features representing heterogeneous LF tasks](image-url)
B. Selection of Candidate LF Models

Many LF models have been developed in literature for solving different LF tasks. In this paper, four LF models commonly used for forecasting distribution system loads are selected: Seasonal Autoregressive Integrated Moving Average (SARIMA), Long Short-Term Memory (LSTM), Support Vector Regression (SVR) and Similar Day (SD). Note that for a given LF model, one can select different model structures in order to achieve the best performance in a given LF task. Therefore, when preparing the LF candidate models, we consider 6 SARIMA and 2 LSTM model structures to demonstrate that the proposed meta-learning framework is also effective in selecting model structures. The 10 candidate LF models are shown in Table I.

| Number | Candidate LF models | Number | Candidate LF models |
|--------|---------------------|--------|---------------------|
| 1      | SARIMA (2,1)        | 6      | SARIMA (5,5)        |
| 2      | SARIMA (3,3)        | 7      | LSTM (125)          |
| 3      | SARIMA (4,2)        | 8      | LSTM (200)          |
| 4      | SARIMA (4,4)        | 9      | SVR                 |
| 5      | SARIMA (5,2)        | 10     | SD                  |

- **SARIMA** models time series with seasonal characteristics [20]. The basic structure of SARIMA is

\[
y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} - \theta_1 e_{t-1} - \theta_2 e_{t-2} - \cdots - \theta_q e_{t-q} \tag{11}
\]

where \( p, q \) determine the structure of the model, and \( \phi, \theta \) are the coefficients. Here we use 6 different structures of SARIMA model, shown in Table I. For example, SARIMA (2,1) refers to the SARIMA model with \( p=2, q=1 \);

- **LSTM** is an upgraded version of Recurrent Neural Network equipped with long-term memory capability [21]. The key structural parameter for LSTM is the number of hidden units. Here we consider 2 typical structures: 125 and 200 hidden units of LSTM.

- **SVR** is a classical regression method by finding a hyperplane to separate high-dimension data [22]. Here we introduce SVR to mainly solve forecasting tasks with exogenous factors. The kernel function we use for SVR is Gaussian kernel.

- **SD** tries to find the most similar day in the historical data [22]. For example, SD can be used when historical load profile is not sufficient for training complex forecasting models. Let \( \Delta T = T_{\text{fore}} - T_{\text{hist}} \), where \( T_{\text{fore}}, T_{\text{hist}} \) represent the time indexes of the forecasting day and the historical day. Then the similarity between the forecasting day and the historical day, \( \gamma \), is calculated as

\[
\gamma = \frac{\beta_1 (\text{mod}(T/7)) + \beta_2 (\text{floor}(T/7)) + \beta_3 (\text{floor}(T/365))}{\|X_j(T_{\text{fore}}) - X_j(T_{\text{hist}})\|_2}
\tag{12}
\]

where \( \beta_1, \beta_2, \beta_3 \in (0,1) \), \( C \) is binary variable that equal 1 when \( \text{mod}(T/365) = 0 \) and equal to 0 otherwise. In (12), the numerator measures the calendar similarity and the denominator quantifies the distance of the exogenous factors, so the historical day with the largest \( \gamma \) will be selected as the SD for the forecasting day.

C. Candidate model labeling

To find the best performed LF model \( \Phi(j) \) for LF task \( j \), (2)-(5) are repeated \( L \) times with different training and testing data splits. This allows us to obtain an estimation of the distribution of the top 1 LF model, \( \Omega_j \). One can iterate the process until the distribution is stabilized. Pearson correlation coefficient [24], \( P_{cc} \), is used as the stopping criterion. \( P_{cc} \) is a statistic that measures the correlation between two vectors. The iteration will stop when \( P_{cc} \) between \( \Omega_j(l) \) and \( \Omega_j(l-1) \) is larger than 0.95. Note that this step is critical for removing the uncertainty in selecting the best LF model for each LF task. The pseudocode of determining \( \Phi(j) \) is shown in Algorithm 1.

Algorithm 1: Candidate model labeling for LF task \( j \)

Input: \( X_j \)

Output: \( \Phi(j) \)

Initialize \( P_{cc} = 0, L = 1 \)

while \( P_{cc} < 0.95 \) do

increase \( L \) by 10

for \( 1, 2, \ldots, L \) do

Randomly split \( X_j \) into \( X_j^{\text{train}} \) and \( X_j^{\text{test}} \)

Train each candidate LF model based on (2)(3)

Test each candidate LF model based on (4)(5)

Label the LF model with the smallest RMSE as top 1

end

Calculate \( \Omega_j(l) \)

Calculate \( P_{cc} \) between \( \Omega_j(l) \) and \( \Omega_j(l-1) \)

end

return \( \Phi(j) \) = LF model with the highest frequency in \( \Omega_j(l) \)

D. Meta-feature of LF Tasks

In this paper, a feature set \( F \) containing 16 features (See Table II) is used to characterize each task. Features 1-6 describe the basic task features of a LF tasks. Features 7-16 are statistics characterizing the historical load profile.

| Number | Features | Number | Features |
|--------|----------|--------|----------|
| 1      | Data length | 9      | Minimum  |
| 2      | Number of weather features | 10     | Standard deviation |
| 3      | Data granularity | 11     | Kurtosis  |
| 4      | Forecasting horizon | 12     | Skewness  |
| 5      | Number of customers | 13     | Fickleness |
| 6      | Load type | 14     | H-ACF     |
| 7      | Mean | 15     | H-PACF    |
| 8      | Maximum | 16     | Periodicity |

Kurtosis and Skewness can be calculated by (13) and (14), where \( \sigma, \bar{y} \) are the standard deviation and mean value of the historical load profile.

\[
\text{Kurtosis} = \frac{1}{\sigma^4} \sum_{n=1}^{N} [y(n) - \bar{y}]^4 \tag{13}
\]

\[
\text{Skewness} = \frac{1}{\sigma^3} \sum_{n=1}^{N} [y(n) - \bar{y}]^3 \tag{14}
\]

Fickleness measures the ratio of a time series crossing its mean value and is calculated as

\[
\text{Fickleness} = \frac{1}{2} \sum_{n=2}^{N} \left| \text{sign}[y(n-1) - \bar{y}] - \text{sign}[y(n-1) - y] \right| \tag{15}
\]

Highest Autocorrelation Function (H-ACF) and Highest Partial Autocorrelation function (H-PACF) measure the self-correlation features of the load profile, which is especially useful for determining the structure of a SARIMA model.
**Periodicity** of the load profile is usually related to the data granularity. For example, periodicity is usually 24 or 168 for an hourly load profile and 30 for a daily load profile.

### E. Meta-learner selection

A meta-learner maps the meta-task features \( \mathbf{F} \) to the best LF models \( \Phi \) for a given LF task. This makes it essentially a classification problem. Thus, in this paper, 4 different classification algorithms with different strength in classification are selected: Random Forest (RF) [25], K-Nearest Neighbor (KNN) [26], Naïve Bayesian (NB) [27] and Linear Discrimination (LD) [28].

### F. Scoring-voting Mechanism

To combine the four recommendations from the four meta-learners into one, a scoring-voting mechanism is developed, as shown in Fig. 3.

![Fig. 3. Scoring-voting system to combine different meta-learners.](image)

Note that each classifier accomplishes its classification based on an internal scoring procedure. For example, NB calculates the posterior probability of each class as their scores, while RF counts the voting results from its wrapped decision trees. The meta-learner \( i_M \) selects the candidate model with the highest score \( S_{im} \) as its output. A higher score means a stronger belief of the classifier on its output, therefore leading to a higher classification accuracy.

We then establish the relationship \( h_{im} \) between the score \( S_{im} \) and the classification accuracy \( \eta_{im} \) for each meta-learner, based on their performance on the testing LF tasks

\[
\eta_{im} = h_{im}(S_{im}) \quad (16)
\]

When dealing with a new task online, we first transfer the scores \( S_{im} \) of each meta-learner to their accuracy level \( \eta_{im} \) using (16), and then select the candidate LF model with the highest accuracy level as the final choice \( \Phi^{\text{new}} \).

### IV. Experiment Setup and Results

This section presents experiments and results for evaluating the performance of the meta-learning based distribution LF model selection framework.

### A. LF Task Setup

Creating a large amount of heterogeneous LF tasks is crucial for training the meta-learner. As summarized in Table III, we consider five key forecasting requirements: the aggregation level, the number of weather features, the historical data lengths, forecasting horizon, and granularity of data, each of which represents one of the five dimensions shown in Fig. 3.

![Table III: Heterogeneity in forecasting task selection](image)

### B. LF Model Selection in the Base-learning Layer (All task)

Residential and commercial load profiles are from two data sources: 15-minute and 30-minute smart meter data sets collected from utilities in North Carolina areas and 1-minute data sets from Pecan Street data repository [29]. Hourly weather data is downloaded from the National Oceanic and Atmospheric Administration (NOAA) website [30]. In this paper, 846 LF tasks are constructed by the exhaustive combination of the LF requirements within the ranges given in Table III and by following the following additional considerations:

- From the building-level to the feeder-level, LF tasks are designed for two main classes of loads: residential and commercial. Industrial load and agriculture loads are not considered.
- From the building-level to the microgrid-level, we focus on short-term LF because the goal of such LF tasks is usually to support demand response management programs in operation [31].
- At the feeder-level, mid-term LF tasks are also considered. Also, we assume that we have up to 12 available weather features from weather service providers.
- In practice, historical data available for a distribution LF task can be very short so we considered three cases to cover the data availability issue: 1-month, 6-month and 1-year. Also, because weather data may not always be available in a distribution LF task, we consider the case with zero weather feature.

### B. LF Model Selection in the Base-learning Layer (all task)

Following the LF model selection process introduced in Section III, the statistically best performed LF models for 846 LF tasks are selected. In Fig. 4(a), we use the result from one of the 846 tasks as an example to illustrate the best LF model selection process. When the number of iteration increases, the distribution of the top 1 LF model starts to stabilize. After 60 iterations, the Pearson coefficient is above 0.98, so the iteration stops and model 9 is selected as the best performed model for this task. The box plot in Fig. 4(b) shows that approximately 50% of the 846 LF tasks require 20 to 40 iterations to identify the best LF model using the proposed method.
The model selection results of all the 846 LF tasks are summarized in Fig. 5. Note that if the historical data of a LF task is insufficient to train a LF model, we consider the LF model as an infeasible model for accomplishing this LF task.

![Fig. 4. (a) An example of the LF model selection process in the base-learning layer. (b) boxplot of required iterations to label each of the 846 LF tasks.](image)

**Fig. 4.** (a) An example of the LF model selection process in the base-learning layer. (b) boxplot of required iterations to label each of the 846 LF tasks.

![Fig. 5. LF model selection results on all 846 tasks.](image)

**Fig. 5.** LF model selection results on all 846 tasks.

The results show that model 7, LSTM (125), is the most frequently-selected model (173 out of 846 tasks) and model 10, the SD model, has by far the shortest training time and lowest data requirement among all options, but the mean and variance of its forecasting error are larger than other LF models.

Because the SARIMA-based approach requires significant amount of historical data to train, SARIMA-based LF models have higher failure counts. Among the six SARIMA models selected, SARIMA (5,5) (model 6) failed in 64 tasks and SARIMA (4,4) (model 4) failed in 40 tasks. However, on average, they tend to have a higher forecasting accuracy for the tasks with sufficient training data sets.

To further quantify the distance between different LF models, we define the System Error Ratio (SER) as

$$\text{SER} = \frac{E_{\text{select}}}{E_{\text{best}}}$$

where $E_{\text{select}}$ is the forecasting RMSE of a selected candidate model on a specific task, and $E_{\text{best}}$ is the forecasting RMSE of the actual best model among the candidates on this task. SER measures the distance between the selected model and the actual best model for each LF task.

**C. LF Tasks Similarity Evaluation**

Recall that at the meta-learning level, the input feature matrix of the meta-learning layer, $\mathbf{F}$, consists of two parts: input data statistics and technical requirements of the LF task. We apply T-distributed Stochastic Neighbor Embedding (t-SNE) [32] to visualize the similarity among the LF tasks. Through nonlinear dimension reduction, the originally 16-dimensional $\mathbf{F}$ is reduced to a two-dimension matrix so distancing-based clustering method can be used to identify the five representative clusters, as shown in Fig. 7 and Table IV.

![Fig. 6. SER values of candidate LF models under different rankings.](image)

**Fig. 6.** SER values of candidate LF models under different rankings.

![TABLE IV](table)

| Cluster | Load Level | Weather Feature | Historical Data Length (day) | Forecasting Horizon (h) | Data Granularity (h) |
|---------|------------|-----------------|------------------------------|------------------------|---------------------|
| 1       | 1000-2000  | 0.1-12          | 30, 180, 360                 | 720                    | 24                  |
| 2       | 1000-2000  | 0.1-12          | 30                           | 4-168                  | 24                  |
| 3       | 1-10       | 0.1             | 30                           | 4-24                   | 0.25-1              |
| 4       | 50-300     | 0.1             | 30, 180, 360                 | 4-168                  | 0.25-1              |
| 5       | 1500-2000  | 0.1-12          | 180, 360                     | 4-168                  | 1                   |

In Fig. 7, each colorized dot represents a LF task labeled by its best performance model. After applying t-SNE visualization, similar LF tasks are more likely to appear near each other whereas dissimilar LF tasks appear far apart with each other. The results show that clusters 1, 2, and 5 represent feeder-level LF tasks, among which cluster 1 represents mid-term forecasting with daily data granularity with SD as the dominant model. This is because although historical data is insufficient to

![Fig. 7. t-SNE visualization of LF tasks labeled by their best performance models](image)

**Fig. 7.** t-SNE visualization of LF tasks labeled by their best performance models.
train a complicate model in those cases, the load profile is normally stable and exhibits clear periodicity. Clusters 2 and 5 represent short-term forecasting with weather features available, making LSTM the best model in most cases. Cluster 3 represents short-term LF tasks at the building- and transformer-levels, where the load profiles are highly volatile, making SVR more frequently picked. Cluster 4 represents short-term microgrid-level LF tasks with little weather information, making LF tasks more often formulated as time series analysis problems suitable for SARIMA.

D. Meta-learner Training, Validation, and Testing Results

To train, validate, and test the LF model selection meta-learner, we randomly split 846 tasks into three groups: training (70%), validation (20%), and testing (10%), respectively. After the four meta-learners are trained on the training set, their performance will be validated on the validation set. Then, LF tasks will be randomly selected from the testing set to evaluate the performance for a meta-learner to provide online LF model recommendation (see next subsection E).

Figure 8 shows the validation results on LF tasks in cluster 4 as an example. Each blue dot represents a validation LF task in cluster 4. If a meta-learner successfully identifies the best LF model for a LF task, a circle of its specified color will be placed around the dot. Thus, “no circle around a dot” means that none of the four meta-learners identified the best model; “multiple circles around a dot” means that more than one meta-learners have successfully identified the best LF model.

As discussed in Section IX, to value the strength of each meta-learner, a scoring-voting mechanism is developed to weight recommendations from each meta-learner. As shown in Fig. 9, in general, the score $S$ is proportional to the meta-learner classification accuracy calculated by (9). This allows the recommendation with the highest score to be used as the final recommended model. As shown in Table V, the classification accuracy of the proposed scoring-voting mechanism is 46%, which is 36% higher than the baseline (random selection) and 5-13% higher than that of an individual meta-learner. Finally, as shown in Table VI, the proposed scoring-voting meta-learning mechanism can significantly reduce the forecasting error compared to any single LF model as well as successfully avoid selecting LF models that cannot perform the LF task.

In addition to recommending the best LF model for a LF task, the proposed meta-learning framework can also rank all candidate models so that the second-best or third-best LF models can be recommended. As shown in Table VII, the average SER of the top three models are all lower than the single models listed in Fig. 5, whereas the classification accuracy of the three models are all above the baseline 10% (a sum of accuracy is 76%) with little or no failure. This means the meta-learning system can recommend on average three high-quality LF models for each LF task.

E. Online Testing Performance

Two LF tasks, with features presented in Table VIII, are used to illustrate the online operation procedure. Task 1 is a transformer-level short-term (24 hours ahead) LF task with 30 days, 15-minute historical load data; Task 2 is a feeder-level short-term (1-week ahead) LF task using 6-month, 1-hour historical data. The meta-features for the two tasks are first calculated and input to the trained meta-learner to obtain the recommended model. The top one model is recommended in task 1 and the top three model are recommended in task 2.

Results are shown in Fig. 10 and table IX. We can see that for task 1 the system successfully recommends the actual best model SD. For task 2, three SARIMA models with similar performance are recommended as the top 3, with the actual best model SARIMA(5,5) ranked as the second. Note that the most time-consuming part of the proposed meta learning system is labeling candidate models, where all candidate models are executed for completing all the sample tasks exhaustively. However, this part can be done once for all during offline training. The online procedure is simply a numerical calculation of task features and a forward application of the trained meta learner, which is very efficient and thus can
be deployed in distributed devices. Users can specify a threshold for triggering system updates to add new forecasting tasks to the training set and retrain the meta learner. The main advantage of applying such a highly extendable system is that users can always add new features and candidate LF models to the meta-learning framework and avoid updating the whole system from scratch.

Fig. 10. (a) LF results of Task 1, (b) LF results of Task 2.

| TABLE IX | RECOMMENDATIONS AND FORECASTING ACCURACY IN ONLINE APPLICATION |
|----------|---------------------------------------------------------------|
|          | Task 1 | Task 2 |
| Actual best model | SD    | SARIMA(5,5) |
| Top-1 Recommendation & MAPE | SD, 19% | SARIMA(2,2), 8% |
| Top-2 Recommendation & MAPE | SARIMA(5,5), 6% | SARIMA(4,4), 6% |

V. CONCLUSION

In this paper, we presented a meta-learning based LF model selection framework for handling heterogeneous forecasting tasks in distribution networks. Each meta-learner will learn to select the LF model with the best performance for a given LF task in the offline training. The score-voting mechanism will learn to weight recommendations from different meta-learners based on their strength in identifying the top candidate models. The resultant system recommends on average up to three effective LF models for each given LF task. Simulation results show that the top one recommendation has 46% chance and the top three recommendations have 76% chance to identify the actual best LF model. The mechanism is highly scalability and extendibility because it allows users to introduce new features or candidate models. Our future work will focus on the feature engineering to further improve the model selection accuracy.

REFERENCES

[1] M. S. Kandil, S. M. El-Deibeik, and N. E. Hasanian, “Long-term load forecasting for fast developing utility using a knowledge-based expert system,” IEEE Trans. on Power Syst., vol. 17, no. 2, Aug. 2002.
[2] M. S. Kandil, S. M. El-Deibeik, and N. E. Hasanian, “The implementation of long-term forecasting strategies using a knowledge-based expert system: part-II,” Electric Power Systems Research, vol. 58, no. 1, pp. 19-25, May 2001.
[3] S. M. R. Kazemi, M. M. Seied Hoseini, S. Abbassian-Naghne, and S. H. A. Rahmati, “An evolutionary-based adaptive neuro-fuzzy inference system for intelligent short-term load forecasting,” International transactions in operational research, vol. 21, no. 2, Mar. 2014.
[4] S. H. Liao, “Expert system methodologies and applications—a decade review from 1995 to 2004,” Expert systems with applications, vol. 28, no. 1, pp. 93-105, Jan. 2005.
[5] S. Thran, L. Pratt, “Learning to learn: Introduction and overview,” in Learning to learn, Boston, MA, Springer, 1998, pp. 3-17.
[6] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” arXiv preprint, arXiv:1703.03400, 2017.
[7] C. Cui, T. Wu, M. Hu, J. D. Weir, and X. Li, “Short-term building energy model recommendation system: A meta-learning approach,” Applied energy, vol. 172, pp. 251-263, Jun. 2016.
[8] M. Feurer, J. T. Springenberg, and F. Hutter, “Initializing bayesian hyperparameter optimization via meta-learning,” in Twenty-Ninth AAAI Conference on Artificial Intelligence, Feb. 2015.
[9] C. Lemke, M. Budka, and B. Gabrys, “Meta-learning: a survey of trends and technologies,” Artificial intelligence review, vol. 44, no. 1, pp. 117-130, Jun. 2015.
[10] M. Matijai, J. A. Suykens, and S. Krajcar, “Load forecasting using a multivariate meta-learning system,” Expert systems with applications, vol. 11, pp. 4427-4437, Sep. 2015.
[11] A. Arjit, R. Samizadeh, and M. D. Sarayzad, “Meta-learning in multivariate load demand forecasting with exogenous meta-features,” Energy Efficiency, pp. 1-17, Feb. 2020.
[12] X. Wang, K. Smith-Miles, and R. Hyndman, “Rule induction for forecasting method selection: Meta-learning the characteristics of univariate time series,” Neurocomputing, vol. 72, no. 10-12, pp. 2581-2594, Jun. 2009.
[13] T. S. Talagala, R. J. Hyndman, and G. Athanasopoulos, “Meta-learning how to forecast time series,” Monash Econometrics and Business Statistics Working Papers, vol. 6, pp.18, Apr. 2018.
[14] C. Lemke, and B. Gabrys, “Meta-learning for time series forecasting and forecast combination,” Neurocomputing, Vol. 73, no. 10-12, pp. 2006-2016, Jun. 2010.
[15] J. Hu, J. Heng, J. Tang, and M. Guo, “Research and application of a hybrid model based on Meta learning strategy for wind power deterministic and probabilistic forecasting,” Energy Conversion and Management, vol. 173, pp. 197-209, Oct. 2018.
[16] J. R. Rice, “The algorithm selection problem,” In Advances in Computers, vol. 15, Elsevier, 1976, pp. 65-118.
[17] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, “Meta-learning in neural networks: A survey,” arXiv preprint, arXiv:2004.05439, 2020.
[18] S. Ali, and K. A. Smith-Miles, “A meta-learning approach to automatic kernel selection for support vector machines”. Neurocomputing, vol. 70, no. 1-3, pp. 173-186, Dec. 2006.
[19] C. M., Lee, and C. N. Ko, “Short-term load forecasting using lifting scheme and ARIMA models,” Expert Systems with Applications, vol. 38, no. 5, pp. 5902-5911, May, 2011.
[20] T. Fang, and R. Lahdelma, “Evaluation of a multiple linear regression model and SARIMA model in forecasting heat demand for district heating system,” Applied energy, vol. 179, pp. 544-552, Oct. 2016.
[21] W. Kong, Z. Dong, D. J. Hill, F. Luo, and Y. Xu, “Short-term residential load forecasting based on resident behavior learning,” IEEE Trans. Power Syst., vol. 33, no. 1, pp. 1087-1088, Mar. 2017.
[22] Y. Chen, P. Xu, Y. Chu, W. Li, Y. Wu, L. Ni, Y. Bao, and K. Wang, “Short-term electrical load forecasting using the Support Vector Regression (SVR) model to calculate the demand response baseline for office buildings,” Applied Energy, vol. 195, pp. 659-670, 2017.
[23] Y. Chen, P. B. Luh, C. Guan, Y. Zhao, L. D. Michel, M. A. Coolbeth, P. B. Friedland, and S. J. Roucke, “Short-term load forecasting: Similar day-based wavelet neural networks,” IEEE Trans. Power Syst., vol. 25, no. 1, pp. 322-330, Nov. 2009.
[24] J. Benesty, J. Chen, Y. Huang, and J. Cohen, “Pearson correlation coefficient,” In Noise reduction in speech processing, Springer, Berlin, Heidelberg, 2009, pp. 1-4.
[25] A. Lahouar, and J. B. H. Slama, “Day-ahead load forecast using random forest and expert input selection,” Energy Conversion and Management, vol. 103, pp. 1040-1051, Oct. 2015.
[26] W. Gao, S. Oh, and P. Viswanath, “Demystifying Fixed k-Nearest Neighbor Information Estimators,” IEEE Trans. Info. Theory, vol. 64, no. 8, pp. 5629-5661, Feb. 2018.
[27] X. Z. Wang, Y. L. He, and D. D. Wang, “Non-naive Bayesian classifiers for classification problems with continuous attributes,” IEEE Trans. Cyber., vol. 44, no. 1, pp. 21-39, Feb. 2013.
[28] A. Tharwat, “Linear vs. quadratic discriminant analysis classifier: a tutorial,” International Journal of Applied Pattern Recognition, vol. 3, no. 2, pp. 145-180, Sep. 2016.
[29] Pecan Street Dataport, 2020. [Online]. Available: https://www. pecanstreet.org/dataport/
[30] National Oceanic and Atmospheric Administration, 2020. [Online]. Available: https://www.noaa.gov.
[31] Q. Zhou, M. Shahidehpour, A. Paaso, S. Bahramirad, A. Abdulwahab, and A. M. Abusorrah, “Distributed Control and Communication Strategies in Networked Microgrids,” IEEE Commun. Surveys Tuts., early access, 2020, doi: 10.1109/COMST.2020.3023963.
[32] L. V. D. Maaten, and G. Hinton, “Visualizing data using t-SNE,” Journal of machine learning research, vol. 9, pp. 2579-2605, Nov. 2008.