Few-shot Residential Load Forecasting Boosted by Learning to Ensemble

Jinhao Liang*, Chenbei Lu†, Wenqian Jiang*, Chenye Wu*
*School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, Guangdong 518172, China
†Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing 100084, China
jinhaoliang1@link.cuhk.edu.cn, lcb20@mails.tsinghua.edu.cn, wenqianjiang@link.cuhk.edu.cn, chenyewu@yeah.net

***Abstract*—Probabilistic forecasting can characterize the uncertainties and the dynamic trends of the future residential load, while massive data are required for popular forecasting methods. In this study, we consider probabilistic load forecasting for residential users who are only willing to provide limited data samples due to privacy concerns. To address this challenge, we analyze the characteristics of residential load and employ clustering-based few-shot learning methods to augment the data. Meanwhile, we combine different models, known as model ensemble, to further improve the performance. Compared with conventional ensemble methods using the linear combination, we adopt learning to ensemble, which captures the strengths of various models by learning the optimal nonlinear combination to avoid performance loss. We demonstrate that the proposed method outperforms conventional rivals theoretically and empirically. This method also sheds light on how varying the number of provided data can accommodate different privacy concerns.**

***Index Terms*—Residential Load Forecasting, Few-Shot Learning, Learning to Ensemble**

I. Introduction

Probabilistic residential load forecasting (PRLF) plays a crucial role in the emerging active distribution grid, as it provides comprehensive statistics of the future load to facilitate decision-making [1]. For instance, home energy management aims to control the flexible load according to forecast information to minimize the total cost [2]. Over the past several decades, various probabilistic forecasting methods have demonstrated remarkable performance based on sufficient data. However, privacy concerns may deter residential users from providing the necessary data to service providers for future load prediction. Consequently, there is a pressing need to develop PRLF methods that are capable of delivering high performance using few shot data samples or even in the absence of data samples at all.

A. Challenges and Opportunities

The first challenge in few-shot PRLF arises from insufficient data. Classical data augmentation methods, such as rotation and cropping, cannot be incorporated into load data due to their temporal dependency. To address this issue, various time series data augmentation methods have been proposed, including noise injection and window warping [3]. However, these methods often overlook the specific characteristics of the task at hand, resulting in inefficient data augmentation. To overcome this shortcoming, we analyze the characteristics of the residential load and propose a clustering-based few-shot learning (FSL) method. This approach aims to identify the target user's load pattern and augment scarce data using similar patterns.

The second challenge involves leveraging the potential of augmented data effectively. While a number of models have demonstrated remarkable performance in PRLF, our observation indicates that no single model consistently outperforms others across all scenarios. Such an observation suggests that a model ensemble is required to achieve consistently improved forecasting performance. However, classical ensemble methods, such as mean value and weighted average ensemble, are essentially linear combinations, which may limit the forecasting performance. To this end, we propose a learning to ensemble approach designed for few-shot PRLF. This approach captures the optimal nonlinear combination of single models, resulting in improved performance over rivals.

Both theoretically and empirically, we demonstrate that our proposed methods surpass its rivals. The whole process is visualized in Figure 1.

B. Related Works

Few-shot load forecasting has been studied to address the challenges brought by few shot data samples. Lee *et al.* utilize transfer learning and meta learning to develop a high-performance individualized forecasting model using limited target user data, supplemented by extensive additional data in [4]. Wu *et al.* introduce an attentive transfer framework to ensemble the Graph Neural Network (GNN) models trained on

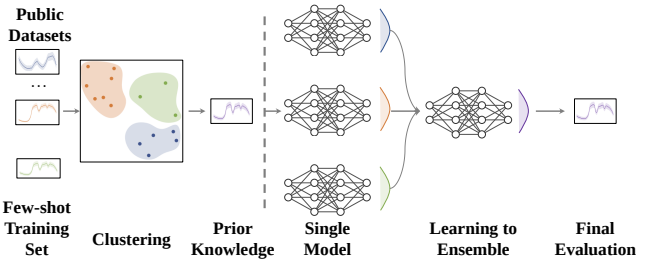


Fig. 1. Algorithm Framework.

C. Wu is the corresponding author. This work was supported by the National Natural Science Foundation of China under Grant 72271213, and the Shenzhen Science and Technology Program under Grant JCYJ20220530143800001 and Grant RCYX20221008092927070.

the target user data and the GNN model trained on additional data in [5]. Notably, these studies do not account for the similarity between target user data and additional data, which may compromise the performance. In [6], Lu *et al.* address this issue by transferring the historical load data of other users in the same area to supplement data samples. Different from this heuristic method, Wang *et al.* design a clustering method that leverages prior knowledge from clustering results to forecast loads with limited data in [7]. We further this line of research by employing a clustering-based few-shot learning approach. In this paper, we offer theoretical insights into this method, and demonstrates its remarkable performance with few shot data samples.

There is also a growing body of literature on model ensemble. Leandro *et al.* propose a modified Passive Aggressive Regression model to implement the online ensemble forecasting in [8]. Wang *et al.* introduce a constrained quantile regression averaging method to integrate several individual probabilistic forecasts in [9]. Nonlinear ensemble methods have also been investigated recently. In [1], Lu *et al.* develop a learning ensemble framework capable of capturing the optimal nonlinear combination of different single models for probabilistic load forecasting. However, these works mainly assume the availability of sufficient data, a condition that is impractical in our setting. Different from these studies, we highlight the benefits of employing neural networks to construct nonlinear probabilistic load forecasting model ensembles from both theoretical and empirical perspectives.

II. Problem Formulation

PRLF offers a significant improvement over traditional load forecasting methods by predicting not only the expected values but also the distribution of future loads [2]. Specifically, probabilistic forecasting can be transformed into quantile forecasting, guided by pinball loss as follows:

$$\mathcal{L}(\hat{d}, d, \theta) = \begin{cases} (1-\theta)(\hat{d}-d) & \hat{d} \geq d, \\ \theta(d-\hat{d}) & \hat{d} < d, \end{cases} \tag{1}$$

where d and $\hat{d}$ denote the actual load and predicted load, respectively. And θ is the pre-determined quantile. Given a probabilistic load predictor at time T, its inputs are the historical loads for the previous K^{in} time slots, defined as $\boldsymbol{d}_T^{\text{in}} = [d_{T-K^{\text{in}}+1}, ..., d_T]^{\mathsf{T}}$. Considering a target set of quantiles $\boldsymbol{\Theta} = \{\theta_1, ..., \theta_Q\}$, where Q is the number of target quantiles, the PRLF $\mathcal{F}$ aims to forecast the loads $\hat{\boldsymbol{D}}_T^{\text{out}}$ of each quantile in the upcoming K^{out} time slots. Mathematically:

$$\hat{\boldsymbol{D}}_T^{\text{out}} = \mathcal{F}(\boldsymbol{d}_T^{\text{in}}), \tag{2}$$

where $\hat{\boldsymbol{D}}_T^{\text{out}} = [\hat{\boldsymbol{d}}_{T,\theta_1}^{\text{out}}, ..., \hat{\boldsymbol{d}}_{T,\theta_Q}^{\text{out}}]$. Each $\hat{\boldsymbol{d}}_{T,\theta_i}^{\text{out}} = [\hat{d}_{T+1,\theta_i}^{\text{out}}, ..., \hat{d}_{T+K^{\text{out}},\theta_i}^{\text{out}}]$, where $\hat{d}_{T+k,\theta_i}^{\text{out}}$ denotes the predicted load at time $T+k$ with percentile θ_i.

The major challenge in PRLF is the requirement of large volume data, which may raise users' concerns about privacy leakage. As we mentioned, this motivates us to study the few-shot PRLF. To mathematically highlight the challenges in this task, we introduce the following concepts.

Consider a dataset of input-output pairs $(\mathcal{X}, \mathcal{Y})$. $\mathcal{X}$ is the set of all inputs, i.e., $\mathcal{X} = \{\boldsymbol{d}_i^{\text{in}}, \forall i\}$. The set of the corresponding output ground truth is $\mathcal{Y} = \{\boldsymbol{d}_i^{\text{out}}, \forall i\}$. $\{\boldsymbol{d}^{\text{in}}, \boldsymbol{d}^{\text{out}}\}$ forms an input-output pair. Few-shot PRLF aims to minimize the expected risk (or generalization error) R for a specific hypothesis $h : \mathcal{X} \to \mathcal{Y}$ [1], which is the loss measured with respect to $P(\mathcal{X}, \mathcal{Y})$ implying the ground truth joint probability distribution of input $\mathcal{X}$ and output $\mathcal{Y}$ [10]. Specifically,

$$R(h) = \int \mathcal{L}(h(\mathcal{X}), \mathcal{Y}, \theta)\ dP(\mathcal{X}, \mathcal{Y}) = \mathbb{E}[\mathcal{L}(h(\mathcal{X}), \mathcal{Y}, \theta)]. \tag{3}$$

Since $P(\mathcal{X}, \mathcal{Y})$ is unknown, the empirical risk $R_I(h)$ serves as a proxy for $R(h)$

$$R_I(h) = \frac{1}{I} \sum\nolimits_{i=1}^{I} \mathcal{L}(h(\boldsymbol{d}_i^{\text{in}}), \boldsymbol{d}_i^{\text{out}}, \theta), \tag{4}$$

where I represents the sample size of few-shot training dataset $D_{\text{fs-train}}$. Accordingly, we define

- $\hat{h} = \arg\min_h R(h)$ as the function that minimizes the expected risk;
- $h_I = \arg\min_{h \in \mathcal{H}} R_I(h)$ as the function in $\mathcal{H}$ that minimizes the empirical risk.

Since $\hat{h}$ is unknown, it must be approximated by some $h \in \mathcal{H}$. h_I represents the optimal hypothesis in $\mathcal{H}$, obtained via empirical risk minimization. While this method is generally effective with sufficient data, it becomes unreliable in few-shot PRLF. This is because, with limited data, the empirical risk $R_I(h)$ may be far from being a good approximation of the expected risk $R(h)$ [10].

Mathematically, we seek to minimize the total error $\mathbb{E}[R(h_I) - R(\hat{h})]$. We observe this error is influenced by both I (determined by the dataset) and $\mathcal{H}$ (determined by the model). Hence, we propose to utilize clustering-based FSL to augment the dataset. Then, we deploy learning to ensemble to determine a hypothesis space $\mathcal{H}$, which can further reduce the distance between the h_I and $\hat{h}$.

III. Clustering-based Few-shot Learning

In this section, we explain why clustering-based FSL approach is efficient in augmenting the training dataset for PRLF.

Classical data augmentation techniques mainly employ transformations such as rotation and cropping, which cannot be straightforwardly applied to datasets with strong temporal dependencies [3]. Nonetheless, such temporal dependencies can be a blessing. The principal characteristic of the residential load is that the number of underlying daily energy consumption patterns is limited [7]. In practice, we have an abundant public residential load dataset [11], denoted by $\mathcal{D}_{\text{public}}$, and a limited set of load data to target users $\mathcal{D}_{\text{fs-train}}$. Hence, we propose to identify the consumption pattern of a target user

[1] h denotes a function mapping the input $\mathcal{X}$ and output $\mathcal{Y}$ within the hypothesis space $\mathcal{H}$.

in the small dataset $\mathcal{D}_{\text{fs-train}}$ with the help of public dataset $\mathcal{D}_{\text{public}}$ by employing clustering techniques. Specifically, We first cluster $\mathcal{D}_{\text{public}}$ to obtain the clustering model $\mathcal{F}^{\text{cluster}}$. And each subgroup represents a unique load pattern comprising similar load profiles. By averaging all historical data in the same subgroup, we produce a prior knowledge dataset $\mathcal{D}_{\text{pk}}$ for each cluster. Specifically, this paper utilizes the popular clustering method k-means to acquire the proper subgroups. By applying this approach, we derive the augmented dataset $\mathcal{D}_{\text{pk}}$ as a source of prior knowledge:

$$\begin{aligned} \mathcal{S} &= \mathcal{F}^{\text{cluster}}(\mathcal{D}_{\text{fs-train}}), \\ \mathcal{D}_{\text{pk}} &= \frac{1}{n}\sum\nolimits_{i\in\mathcal{S}} \boldsymbol{d}_i^{\text{public}}, \end{aligned} \tag{5}$$

where $\mathcal{S}$ represents the indices of households within the same clusters as $\mathcal{D}_{\text{fs-train}}$, and n denotes the size of $\mathcal{S}$.

Remark 1: In practice, users can identify the load pattern type without providing data to the service provider. This is feasible because inference demands fewer computational resources than training, allowing users to perform $\mathcal{F}^{\text{cluster}}$ by themselves. After identifying the load pattern type, the service provider can offer a corresponding predictor for the target users. Additionally, users can choose to provide few shot data samples, such as a single day's load, to the service provider to further enhance performance.

IV. Learning to Ensemble

After augmenting the dataset, the remaining hurdle is to improve the forecasting performance. We adopt model ensemble, which refers to the process of generating multiple single models that are subsequently combined to form a more powerful model with enhanced performance. Specifically, four popular deep learning models are prepared for learning to ensemble, including Fully Connected Deep Neural Network (FCDNN) [1], Recurrent Neural Networks (RNN) [12], Long Short-term Memory (LSTM) [2], and Gated Recurrent Units (GRU) [13].

We first specify the dataset. Consider the dataset $\mathcal{D}_{\text{pk}}$ which may not contain the historical data of target user. This dataset is equally divided into two parts. The initial portion is employed to train various single forecasting models and can be further subdivided into the training set $\mathcal{D}_{\text{pk}}^t$ and the validation set $\mathcal{D}_{\text{pk}}^v$, at a commonly accepted ratio of $4:1$ in the literature [14]. Similarly, the other segment of the dataset is partitioned into two sets: the model ensemble dataset $\mathcal{D}_{\text{pk}}^s$ and the ensemble validation dataset $\mathcal{D}_{\text{pk}}^e$, adhering to the same ratio. These are utilized to train and validate the ensemble model, respectively.

For the whole process, multiple single forecasting models, denoted by $\mathcal{F}_i$, are first generated from the training set $\mathcal{D}_{\text{pk}}^t$ and the validation set $\mathcal{D}_{\text{pk}}^t$. Subsequently, an ensemble model $\mathcal{G}$ is trained based on model ensemble set $\mathcal{D}_{\text{pk}}^s$ and forecasting models $\mathcal{F}_i$'s:

$$\mathcal{G}(\boldsymbol{d}^{\text{in}}) = G_{ens}(\mathcal{F}_i(\boldsymbol{d}^{\text{in}}), \forall i), \tag{6}$$

where G_{ens} is a mapping function from the predictions of every single model $\mathcal{F}_i$ to the ensemble forecasting results. The performance of ensemble model $\mathcal{G}$ is assessed using the few-shot test set $\mathcal{D}_{\text{fs-test}}^e$, in conjunction with the pinball loss criterion.

Classical ensemble methods such as the mean value ensemble, median value ensemble, and weighted average ensemble linearly combine the outputs of single models. Nevertheless, the assumption of linearity may be too strong. This observation motivates us to explore nonlinear ensemble methods to approximate the real prediction more accurately. Neural networks are capable of overcoming the limitations of linear models by learning the nonlinear relationship between the outputs of single models and ensemble models. Specifically, a neural network denoted by $\mathcal{G}_{ML}$ takes the forecasting results of all N trained models $\boldsymbol{\mathcal{F}} = \{\mathcal{F}_1, ..., \mathcal{F}_N\}$ as inputs and produces the forecasting results $\hat{\boldsymbol{D}}$, as described by

$$\hat{\boldsymbol{D}} = \mathcal{G}_{ML}(\mathcal{F}_i(\boldsymbol{d}^{\text{in}}), \forall i). \tag{7}$$

The loss function $L_{NN}(\mathcal{G}_{ML})$ represents the total pinball loss in the model ensemble training set $\mathcal{D}_{\text{pk}}^s$:

$$L_{NN}(\mathcal{G}_{ML}) = \sum\nolimits_{(\boldsymbol{d}^{\text{in}}, \boldsymbol{d}^{\text{out}})\in\mathcal{S}} L(\boldsymbol{d}^{\text{out}}, \mathcal{G}_{ML}(\mathcal{F}_i(\boldsymbol{d}^{\text{in}}), \forall i), \boldsymbol{\Theta}). \tag{8}$$

Then, the neural network is trained to minimize the pinball loss L_{NN}.

V. Performance Guarantee

In this section, we demonstrate the theoretical performance guarantee of the clustering-based FSL approach and learning to ensemble approach.

A. Sample Size Analysis

We first analyze the effectiveness of clustering-based FSL from a theoretical perspective. Let the sample size of few-shot training set $\mathcal{D}_{\text{fs-train}}$ be $z_{\text{fs-train}}$, and the size of similar training set $\mathcal{D}_{\text{pk}}$ be z_{pk}. Our method effectively enlarges the training data by a factor of $\lambda = \frac{z_{\text{pk}}}{z_{\text{fs-train}}}$. The following theorem demonstrates that our method can effectively enhance the performance of FSL from the perspective of generalization error bounds under two commonly adopted assumptions [15]:

Assumption 1: Consider a set of training samples $\{\boldsymbol{d}_i^{\text{in}}, \boldsymbol{d}_i^{\text{out}}\}$. We assume these samples are drawn $i.i.d.$ from an unknown distribution.

Assumption 2: We assume the activation function is a 1-Lipschitz, positive-homogeneous activation function, which is applied element-wise (such as the ReLU function).

Theorem 1: Given a specific model $\mathcal{F}$ and two hypotheses $h_{\text{fs-train}}$ and h_{pk} that trained with sample sizes $z_{\text{fs-train}}$ and z_{pk}, respectively, the generalization errors of the two hypotheses satisfy:

$$\frac{R(h_{\text{pk}})}{R(h_{\text{fs-train}})} = \sqrt{\frac{z_{\text{fs-train}}}{z_{\text{pk}}}} = \frac{1}{\sqrt{\lambda}}. \tag{9}$$

Proof: We prove Theorem 1 in two steps. We first investigate the generalization error bound. Then, we analyze

the role of sample size in generalization error bound. The key is to use Theorem 1 in [15], which suggests:

$$R(h_{\text{fs-train}}) = \frac{B(\sqrt{2\log(2)d+1})\prod_{j=1}^{d} M_F(j)}{\sqrt{z_{\text{fs-train}}}}, \quad (10)$$

$$R(h_{\text{pk}}) = \frac{B(\sqrt{2\log(2)d+1})\prod_{j=1}^{d} M_F(j)}{\sqrt{z_{\text{pk}}}}, \quad (11)$$

where B is a constant, d implies the depth of the neural network, and $M_F(j)$ represents the upper bounds of Frobenius norms in each layer.

Eqs. (10) and (11) immediately imply our results. ∎

This theorem is significant, as it indicates that the generalization error bound decreases with increasing sample size at the rate of $\sqrt{\lambda^{-1}}$. The generalization error bound approach 0 if λ is very large. The final hurdle is to develop a powerful model that can provide a proper hypothesis space $\mathcal{H}$.

B. Hypothesis Space Analysis

An appropriate hypothesis space $\mathcal{H}$ indicates that we can find a hypothesis $h \in \mathcal{H}$, which can perfectly predict future loads. Hence, we examine how substantially the pinball loss can be reduced through our proposed approach.

Consider a targeted quantile θ and one basic forecasting model $\mathcal{F}$. We adopt the two-layer neural network for model ensemble to predict the next K time steps load. Given inputs $\boldsymbol{d}^{\text{in}}$, the single model's forecasting result is $\hat{\boldsymbol{d}}^{\text{out}} = \mathcal{F}(\boldsymbol{d}^{\text{in}})$. An optimal forecasting model $\mathcal{F}^*$ can accurately predict the true load $\boldsymbol{d}^{\text{out},*}$, i.e., $\boldsymbol{d}^{\text{out},*} = \mathcal{F}^*(\boldsymbol{d}^{\text{in}})$. For the subsequent proofs, we make the following assumption adopted in [1]:

Assumption 3: There exists a Lipschitz-continuous injective mapping function h^* from $\hat{\boldsymbol{d}}^{\text{out}}$ to $\boldsymbol{d}^{\text{out},*}$, i.e., $\boldsymbol{d}^{\text{out},*} = h^*(\hat{\boldsymbol{d}}^{\text{out}})$ for all $\boldsymbol{d}^{\text{in}}$.

The following theorem shows that the neural network can efficiently approximate the optimal hypothesis h^*:

Theorem 2: Given a hypothesis space $\mathcal{H}^{\text{NN}}$ corresponding to a two-layer neural network with ReLU activation function and N_{hidden} neurons in the hidden layer, these exists a hypothesis $h^{\text{NN}} \in \mathcal{H}^{\text{NN}}$ satisfying:

$$\left|h^{\text{NN}}(\hat{\boldsymbol{d}}^{\text{out}}) - h^*(\hat{\boldsymbol{d}}^{\text{out}})\right| \leq \frac{2Q_{h^*}(\boldsymbol{d}^{\max} - \boldsymbol{d}^{\min})}{N_{\text{hidden}} - 1}, \ \forall \hat{\boldsymbol{d}}^{\text{out}}, \quad (12)$$

where h^* denotes the optimal hypothesis, Q_{h^*} is the Lipschitz constant for injective mapping function h^*, $\boldsymbol{d}^{\max}$ and $\boldsymbol{d}^{\min}$ are the upper and lower bounds of the loads.

Proof: We first construct the neural network parameters. Denote the weighted vectot as $\boldsymbol{W} = [w_1, w_2, ..., w_{N_{\text{hidden}}}]^\top$, the bias vector as $\boldsymbol{b} = [b_1, b_2, ..., b_{N_{\text{hidden}}}]^\top$. With the ReLU activation function and input $\boldsymbol{d}$, the output of the neuron network $h^{\text{NN}}(\boldsymbol{d})$ satisfies:

$$h^{\text{NN}}(\boldsymbol{d}) = f_{\text{ReLU}}(\boldsymbol{W}\boldsymbol{d} + \boldsymbol{b}) = \max(\boldsymbol{W}\boldsymbol{d} + \boldsymbol{b}, 0). \quad (13)$$

It is clear that $h^{\text{NN}}(\boldsymbol{d})$ is piecewise linear. Now we seek to design the parameters $\boldsymbol{W}$ and $\boldsymbol{b}$ such that, for all integer $k \in [0, N_{\text{hidden}} - 1]$:

$$\begin{aligned} &h^{\text{NN}}\left(\boldsymbol{d}^{\min} + \frac{k(\boldsymbol{d}^{\max} - \boldsymbol{d}^{\min})}{N_{\text{hidden}} - 1}\right) \\ &= h^*\left(\boldsymbol{d}^{\min} + \frac{k(\boldsymbol{d}^{\max} - \boldsymbol{d}^{\min})}{N_{\text{hidden}} - 1}\right). \end{aligned} \quad (14)$$

This condition ensures the neural network can approximate the optimal mapping function h^* in a point-wise manner. Now we will show that the weighted parameter design can meet the requirement of Eq. (14):

$$w_i = \begin{cases} G^i_{h^*}, & i = 1, \\ G^i_{h^*} - \sum_{k=1}^{i-1} w_k, & i > 1, \end{cases} \quad (15)$$

where $G^i_{h^*}$ is a constant satisfying:

$$\begin{aligned} G^i_{h^*} = &\frac{N_{\text{hidden}} - 1}{(\boldsymbol{d}^{\max} - \boldsymbol{d}^{\min})}\left(h^*\left(\boldsymbol{d}^{\min} + \frac{i(\boldsymbol{d}^{\max} - \boldsymbol{d}^{\min})}{N_{\text{hidden}} - 1}\right)\right. \\ &\left. - h^*\left(\boldsymbol{d}^{\min} + \frac{(i-1)(\boldsymbol{d}^{\max} - \boldsymbol{d}^{\min})}{N_{\text{hidden}} - 1}\right)\right). \end{aligned} \quad (16)$$

Additionally, the bias parameter follows:

$$b_i = -w_i\left(\boldsymbol{d}^{\min} + \frac{(i-1)(\boldsymbol{d}^{\max} - \boldsymbol{d}^{\min})}{N_{\text{hidden}} - 1}\right), \forall i. \quad (17)$$

Substituting Eq. (15) and (17) into Eq. (13), we can verify the condition in Eq. (14).

To prove the property of h^{NN}, we follow the same routine of Theorem 4 in [1], which suggests:

$$\left|h^{\text{NN}}(\boldsymbol{d}) - h^*(\boldsymbol{d})\right| \leq \frac{2Q_{h^*}(\boldsymbol{d}^{\max} - \boldsymbol{d}^{\min})}{N_{\text{hidden}} - 1}. \quad (18)$$

where Q_{h^*} is the is the Lipschitz constant for h^*. This concludes our proof. ∎

This is an important theorem that indicates that the ensemble error decreases at the rate of $O(\frac{1}{N_{\text{hidden}}})$. With the increase in the number of neurons N_{hidden}, the neural network can approximate h^* better. When N_{hidden} becomes larger, the ensemble error will approach 0. This theorem indicates that learning to ensemble is capable of providing a hypothesis space $\mathcal{H}$, which includes the optimal hypothesis h^*. Furthermore, this theorem can be easily extended to more complex scenarios following the similar routine of Theorem 19 in [16].

VI. Numerical Studies

In this section, we evaluate the performance of our proposed few-shot ensemble learning framework. All experiments are conducted on a server equipped with an Intel ® i7-9700 CPU, 32 GB RAM, and an NVIDIA ® GeForce RTX™ 3090Ti.

A. Experimental Setup

We extract residential load data for 114 apartments from the UMass Smart Dataset (2017 release) [11] from April 2016 to October 2016, with a resolution of 30 minutes. Specifically, we select 100 apartments as the public dataset to provide cluster results, utilizing the remaining apartments to evaluate the efficacy of our proposed method. The input data are normalized using a common approach. The forecasting outputs consist of PRLF results with percentiles of 20%, 50%, and 80%. The task is to predict load in future intervals of 0.5, 1, 2, and 4 hours. The dataset is split in a 4:1:4:1 ratio, which is used for the training of single models, the validation of single models, the training of the ensemble model, and the validation of the ensemble model, respectively. Specifically, the sample size of the few-shot dataset $\mathcal{D}_{\text{fs-train}}$ includes a load duration of 2 days.

We generate ten RNN, LSTM, and GRU models with 1 or 2 recurrent layers, and the number of features in the hidden state is $16, 32, 64, 128, 256$, respectively. For FCDNN, the neuron count in the first layer is either 16 or 32 or 64, and the second layer has 16 or 32 or 64 or 128 neurons. The FCDNN model used in learning to ensemble has 64 neurons in both layers.

For hyperparameters in each model, we employ random search to determine optimal parameters, including the learning rate and the number of neurons. During the model training process, we utilize the Adam optimizer for optimization and the UP criterion for early stopping, with ReLU functioning as the generic activation function. We adhere to the methods mentioned in classical references for tuning the remaining essential parameters.

B. Competing Methods

We introduce three competing methods:

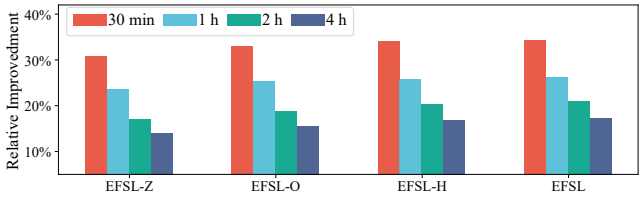
- *Conventional Method (CM)*: We trained a set of 40 single models, as mentioned in Section IV, using the few-shot dataset $\mathcal{D}_{\text{fs-train}}$. Then, we selected the best-performing models as the benchmark.
- *Few-shot Learning Method (FSL)*: Similar to CM, we trained 40 single models using the augmented dataset $\mathcal{D}_{\text{pk}}$. Then, we again selected the best models as the benchmark.
- *Ensemble Few-shot Learning Method (EFSL)*: In this approach, we leveraged learning to ensemble to enhance FSL. Furthermore, we design different strategies with varying numbers of shots used for fine-tuning: 0 shot (EFSL-Z), 1 shot (EFSL-O), half of the few-shot dataset size $\frac{z_{\text{fs-train}}}{2}$ shots (EFSL-H), and $z_{\text{fs-train}}$ shots (EFSL).

C. Performance Evaluation

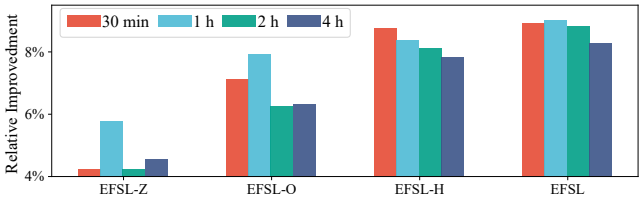
Table I illustrates the pinball loss for different forecasting methods over varying forecasting windows. CM achieves the highest pinball loss across all time intervals. Because training CM with few shots often leads to overfitting, FSL can surpass CM since the risk of overfitting is mitigated by a larger sample size. EFSL consistently outperforms its rivals across all time windows, with the lowest pinball loss. Furthermore,

TABLE I
PERFORMANCE ON UMASS SMART DATASET.

	CM	FSL	EFSL-Z	EFSL-O	EFSL-H	EFSL
30 min	0.4206	0.3036	0.2907	0.2819	0.2770	**0.2765**
1 h	0.3745	0.3034	0.2859	0.2794	0.2781	**0.2761**
2 h	0.353	0.3058	0.2928	0.2866	0.2810	**0.2788**
4 h	0.34	0.3063	0.2924	0.2869	0.2824	**0.2810**



(a) Compared with Best CM.



(b) Compared with Best FSL.

Fig. 2. Performance Evaluation.

the performance of EFSL improves as the number of samples used for fine-tuning increases.

Figure 2(a) illustrates the relative enhancement in the loss of each EFSL model in comparison to the best CM. Notably, even the zero-shot EFSL can improve the benchmark by more than 30% and 10% when the forecasting window is set to 30 min and 4 h, respectively. Moreover, EFSL exhibits substantially superior performance, as the actual load data enable better customization of the specific load pattern through fine-tuning. Specifically, the maximum improvement can exceed 30%. When compared with the best FSL, Figure 2(b) reveals that the EFSL also outperforms the best FSL across all output windows and numbers of shots, with the zero-shot EFSL enhancing the best FSL by over 4% and EFSL enhancing the best FSL by over 8%. Interestingly, this enhancement is nearly 50% when only 1-shot is employed for fine-tuning, and it tends to decrease with the growing number of shots.

D. User-Dependent Performance Analysis

To provide a comprehensive performance comparison of different methods for various residential consumers, Figure 3 characterizes the pinball loss for different methods across different apartments. The x-axis symbolizes the pinball loss for a specific method, while the y-axis represents the pinball loss of EFSL for that apartment. Scatters beneath the line indicate inferior performance compared to EFSL. Most scatters fall below this line, thereby highlighting that our EFSL outperforms other methods across most apartments and forecasting windows. Some zero-shot EFSL points near the line, denoting remarkable performance even in the absence of

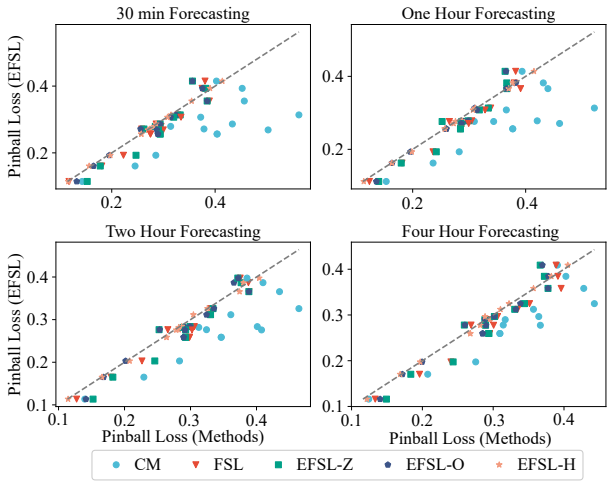


Fig. 3. Performance compassion.

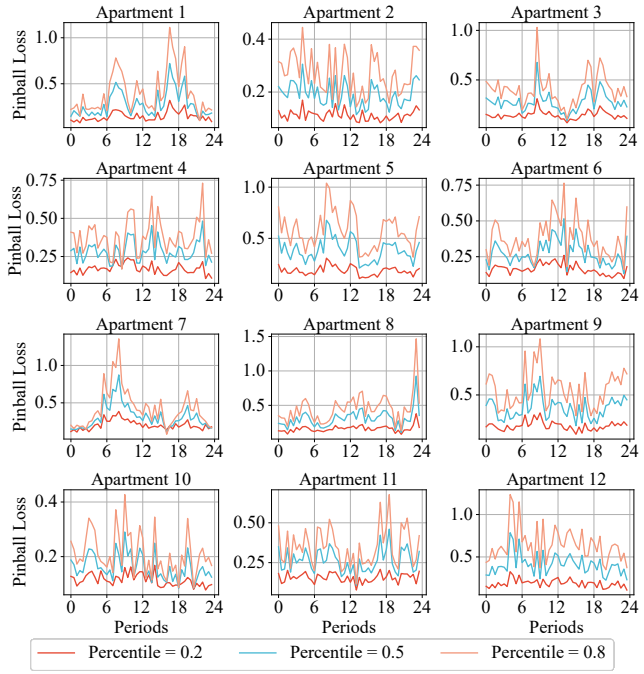


Fig. 4. Pinball Loss Profile.

specific training data. Conversely, most FSL points are situated further from the line, illustrating that the ensemble method significantly bolsters the FSL for most apartments.

E. Temporal Performance Analysis

Figure 4 visualizes the time series of the loss in different percentiles when the forecasting window is 1 hour for some apartments. We can observe the prior knowledge dataset $\mathcal{D}_{\text{pk}}$ may not precisely represent the accurate load pattern for a particular apartment, resulting in a substantial loss when the pattern in $\mathcal{D}_{\text{pk}}$ differs from the actual one. For example, the pinball loss in apartment 8 can be large at about $24:00$. The reason may be that the lifestyle for apartment 8 is much different from the others in its subgroup. Additionally, the pinball loss tends to be higher and more dispersed around $7:00$ and $18:00$, corresponding to the times when people typically wake up and return from work. Consumers have greater load demands and uncertainties during these periods, making the loads more challenging to forecast.

VII. Conclusion

This study employs a clustering-based few-shot learning method and learning to ensemble to enhance the performance of few-shot PRLF. We theoretically and empirically demonstrate the superiority of our proposed approach. Based on our methods, we have designed an appropriate forecasting strategy tailored to various levels of privacy concerns, thereby achieving remarkable performance.

References

[1] C. Lu, J. Liang, W. Jiang, J. Teng, and C. Wu, "High-resolution probabilistic load forecasting: A learning ensemble approach," *Journal of the Franklin Institute*, vol. 360, no. 6, pp. 4272–4296, 2023.

[2] Y. Wang, D. Gan, M. Sun, N. Zhang, Z. Lu, and C. Kang, "Probabilistic individual load forecasting using pinball loss guided lstm," *Applied Energy*, vol. 235, pp. 10–20, 2019.

[3] Q. Wen, L. Sun, F. Yang, X. Song, J. Gao, X. Wang, and H. Xu, "Time series data augmentation for deep learning: A survey," *arXiv preprint arXiv:2002.12478*, 2020.

[4] E. Lee and W. Rhee, "Individualized short-term electric load forecasting with deep neural network based transfer learning and meta learning," *IEEE Access*, vol. 9, pp. 15413–15425, 2021.

[5] D. Wu and W. Lin, "Efficient residential electric load forecasting via transfer learning and graph neural networks," *IEEE Transactions on Smart Grid*, vol. 14, no. 3, pp. 2423–2431, 2023.

[6] Y. Lu, G. Wang, and S. Huang, "A short-term load forecasting model based on mixup and transfer learning," *Electric Power Systems Research*, vol. 207, p. 107837, 2022.

[7] Q. Wang, Z. Chen, and C. Wu, "Clustering enabled few-shot load forecasting," in *2021 IEEE Sustainable Power and Energy Conference (iSPEC)*, (Nanjing, China), pp. 2417–2424, IEEE, 2021.

[8] L. Von Krannichfeldt, Y. Wang, and G. Hug, "Online ensemble learning for load forecasting," *IEEE Transactions on Power Systems*, vol. 36, no. 1, pp. 545–548, 2021.

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

[10] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM computing surveys (csur)*, vol. 53, no. 3, pp. 1–34, 2020.

[11] S. Barker, A. Mishra, D. Irwin, E. Cecchet, P. Shenoy, J. Albrecht, *et al.*, "Smart*: An open data set and tools for enabling research in sustainable homes," *SustKDD, August*, vol. 111, no. 112, p. 108, 2012.

[12] H. Shi, M. Xu, and R. Li, "Deep learning for household load forecasting—a novel pooling deep rnn," *IEEE Transactions on Smart Grid*, vol. 9, no. 5, pp. 5271–5280, 2018.

[13] F. vom Scheidt, X. Dong, A. Bartos, P. Staudt, and C. Weinhardt, "Probabilistic forecasting of household loads: Effects of distributed energy technologies on forecast quality," in *Proceedings of the Twelfth ACM International Conference on Future Energy Systems*, pp. 231–238, 2021.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. http://www.deeplearningbook.org.

[15] N. Golowich, A. Rakhlin, and O. Shamir, "Size-independent sample complexity of neural networks," in *Conference On Learning Theory*, pp. 297–299, PMLR, 2018.

[16] P. Cheridito, A. Jentzen, and F. Rossmannek, "Efficient approximation of high-dimensional functions with neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 7, pp. 3079–3093, 2022.