Unifying Load Disaggregation and Prediction for Buildings with Behind-the-Meter Solar

Yating Zhou, *Student Member*, *IEEE*, and Meng Wang, *Senior Member*, *IEEE*

***Abstract*—Real-time building-level load forecasting is important for demand response and power system planning. Behind-the-meter (BTM) solar generation in buildings is not directly measured, resulting in a lack of native load measurements, even in recorded historical data. This invisibility of native load data makes load forecasting challenging for BTM buildings. Our idea is to learn the unknown and time-varying spatial correlations of nearby buildings to enhance the overall load forecasting accuracy. To the best of our knowledge, this paper, for the first time, integrates load disaggregation and load forecasting without requiring historical native load measurements on BTM consumers. The proposed method, ULoFo, has a computationally efficient load disaggregation component and a state-of-the-art forecasting component. ULoFo also has two interaction strategies, graph sparsification, and input refurbishment, to leverage the intermediate forecasting result to enhance disaggregation accuracy, which in turn further promotes native load forecasting accuracy. ULoFo is demonstrated to outperform existing methods in practical datasets.**

***Index Terms*—Load forecasting, load disaggregation, behind-the-meter solar, spatial-temporal correlations.**

NOMENCLATURE

A. Sets

C^N	Set of buildings with no PV.
C^F	Set of buildings with PV installation and measurements.
C^P	Set of buildings with PV installation but no PV measurement.

B. Parameters

X^N, X^F, X^P	Recorded native load data of C^N, C^F, and C^P in historical T time steps.
S^F, S^P	Recorded PV generations of C^F and C^P in historical T time steps.
N^P	Recorded net load of C^P in historical T time steps.
$\tilde{X}^P, \tilde{S}^P$	Estimated load and PV data of C^P in historical T time steps.
$\tilde{X}^{\text{pattern\_}P}$	Estimated load pattern of C^P in historical T time steps.
$\tilde{S}^{\text{pattern\_}P}$	Estimated PV pattern of C^P in historical T time steps.
$\tilde{\beta}^l, \tilde{\beta}^s$	Estimated load and PV amplitudes of C^P in historical T time steps.
A^G, P^G	Average load and solar adjacency matrices.
$\tilde{A}^G, \tilde{P}^G$	Estimated average load and solar adjacency matrices.
Z^N, Z^F, Z^P	Native load data of C^N, C^F and C^P from time $s-2K+1$ to s.
V^F, V^P	PV generations of C^F and C^P from time $s-2K+1$ to s.
E^P	Net load of C^P from time $s-2K+1$ to s.
$\tilde{Z}^P, \tilde{V}^P$	Estimated load and PV data of C^P from time $s-2K+1$ to s.
A^t_{ij}	Spatial correlation of building i and j at time t.
$\hat{y}^{t+1}$	Predicted load data in all N buildings at time $t+1$.
X^N_t	Native loads of buildings of C^N from time $t-K+1$ to time t.
X^N_{i*}, X^N_{*i}	i-th row of X^N and i-th column of X^N.
r, r^*	Pruning rate and best pruning rate in graph sparsification.
$f(\cdot, \Theta_i)$	i-th forecasting model.
I	The total number of refurbishment iterations.
$A^G_{r^*}, P^G_{r^*}$	The optimal pruned load and solar adjacency matrices.
X, S	Load and solar feature matrices.

Y. Zhou, M. Wang are with the Dept. of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY. Email: {zhouy26, wangm7}@rpi.edu. This research is supported in part by NSF 1932196, the Center for Future Energy Systems (CFES), and the Rensselaer-IBM AI Research Collaboration (http://airc.rpi.edu), part of the IBM AI Horizons Network (http://ibm.biz/AIHorizons).

I. INTRODUCTION

BUILDING-level load forecasting is significant for decision-making, demand management, energy dispatch, and electric transactions in smart grids [1]. Recently, extensive research has been carried out to address the load forecasting problem, but some challenges remain unsolved, such as the invisibility of native load values [2], [3], [4]. This challenge stems from the increasing installation of solar photovoltaics (PV) in buildings [5], [6]. A subset of these PV installations is behind the meter (BTM), primarily due to privacy concerns and the high cost of installing separate smart meters for PV [7]. Consequently, only the net load values (obtained by subtracting PV generations from native load values) are recorded for these buildings, and the corresponding native load values are unobserved [8]. The invisibility of native load values poses challenges for building-level native load forecasting.

Traditional load forecasting methods employ statistical methods such as linear regression [9], exponential smoothing [10], autoregressive moving average (ARMA) [11], and stochastic time series [12]. The forecasting errors of statistical methods, however, increase under abnormal conditions like extreme weather [13] or when the length of the prediction time horizon is long [14]. Machine learning methods can learn complex functions from data and have been applied to load forecasting in recent years [15], [16], [17], [18], [19], [20], [14], [21], [22]. For example, the support vector regression (SVR) method can extract nonlinear and discriminative features [15], but its performance suffers significantly in the presence of data outliers and improper parameter selections [23]. Deep learning models, such as long short-term memory (LSTM) [16], gated recurrent units (GRU) [17], rough autoencoder (RAE) [21], and interval probability distribution learning (IPDL) [22], can capture complicated features and temporal correlations in time series load data and have been applied to the load forecasting problem. Due to the additional spatial correlation in electric consumption patterns [24], [25], spatial-temporal machine learning techniques are proposed to improve the performance of load forecasting [19], [20], [14]. However, all these load forecasting methods require historical native load data as the input data, and none of them can address the challenge of the invisibility of native load values caused by BTM PV generations.

In recent years, very few existing studies have considered native load forecasting with BTM PV. Reference [3] proposes a spatial-temporal graph dictionary learning method for BTM load and PV forecasting, but it requires historical native load and solar data of all consumers. These data may be difficult to obtain from consumers with BTM PV installed. Some studies forecast the net load [26], [27], [28], [29], [30]. Among them, the indirect techniques [26], [27], [28] disaggregate the net load into native load and PV output first, and then forecast native load and PV output separately to finally obtain the aggregated net load forecasting result. The native load forecasting can be viewed as an intermediate step in these approaches. Inspired by these methods, employing load disaggregation techniques can solve the challenge of invisibility of native load values in native load forecasting.

BTM load and PV disaggregation is a process in which net load values can be separated into native load values and PV generations. Existing load disaggregation techniques include model-based approaches [26], [27], [31], [32], [33] and data-driven approaches [2], [4], [28], [34], [35], [36], [37], [38]. Model-based disaggregation approaches usually require accurate physical PV panel models and meteorological data. For example, a physical PV model is established to estimate PV capacity and native load values in [26]. A clean sky model and a PV physical model are combined to disaggregate PV generations and native loads in [32]. One primary disadvantage of model-based solutions is the unavailability of model parameter information[4]. Instead, data-driven approaches have the advantage of using measurement data only and do not rely on physical PV models. In data-driven methods, the correlations and features in measurement data are analyzed and extracted to disaggregate native loads and PV output. For example, reference [2] explores the correlation between monthly nocturnal and diurnal native demands to separate residential customers' loads and PV generations. In [36], a supervised learning model, deep temporal dictionary learning (DTDL), is trained to learn a nonlinear dictionary of energy signals and achieve energy disaggregation. Although these disaggregation methods can estimate historical native loads for BTM consumers and solve the absence of historical data problem, no study analyzes how to prevent disaggregation error propagation when the estimated native load data are used in load prediction. The disjoint processing of disaggregation and forecasting can limit prediction accuracy because disaggregation errors, existing in the estimated input data to the forecasting model, may be propagated into the forecasting process and negatively impact the forecasting performance.

This paper proposes a **U**nified **Lo**ad **Fo**recasting method (ULoFo) for buildings with BTM PV generation. To the best of our knowledge, ULoFo is the *first* method to solve native load disaggregation and prediction jointly and iteratively with *no* requirement of historical native load data on BTM consumers. The existing literature addresses load disaggregation and load forecasting problems separately and often requires historical native load data on BTM consumers. The main idea of ULoFo is to explore spatial correlations of nearby buildings to address the load forecasting challenge caused by BTM solar. ULoFo contains three major parts: (i) the graph smoothing disaggregation model (GSD), (ii) the spatial-temporal attention wavenet prediction model (STAWnet) [39], and (iii) the interaction strategies between GSD and STAWnet. Specifically, the proposed disaggregation model GSD explores the unknown average spatial correlation among native load values and PV generations of nearby buildings to disaggregate the BTM PV and native load values. The state-of-art prediction model STAWnet [39], which has a distinctive advantage over existing load forecasting models in terms of capturing the time-varying spatial-temporal dependence, is applied for native load forecasting by using the measured and disaggregated load as input. Two interaction strategies, including graph sparsification and input refurbishment, are proposed to use the prediction result to prevent disaggregation error propagation and enhance the disaggregation performance, which in turn also boots the prediction performance.

The rest of the content is structured as follows: Section II states the problem formulation of native load forecasting with BTM PV generation. Section III represents the overall framework and details of ULoFo. Section IV conducts numerical experiments, and section V concludes this paper.

II. Problem Formulation and Challenge

A typical native load forecasting problem is that, given real-time native load data, a prediction function, which is learned by historical native loads, can predict future native loads. In the typical native load forecasting scenario, both historical and real-time data are observed. However, the partial historical and real-time native loads are absent due to BTM solar in our setup. Mathematically, our problem formulation is as follows.

Building types: We consider a setup of N nearby buildings that can be divided into three disjoint sets: (i) the set of buildings with no PV, denoted by C^N; (ii) the set of buildings with PV installation and the corresponding smart meters for measuring PV generation, represented by C^F, and (iii) the set of buildings with PV installation but no separate smart meters, represented by C^P. The number of buildings that have solar generation is $M = |C^F| + |C^P|$.

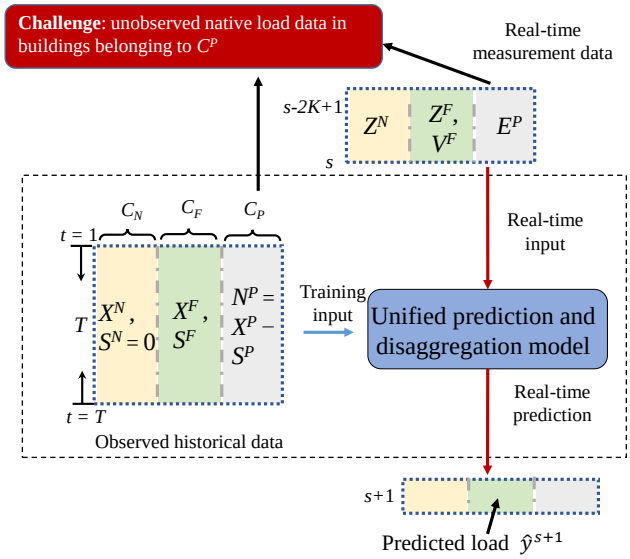

Fig. 1. Native load forecasting with BTM PV

Recorded historical data: Let $X^N \in \mathbb{R}^{T\times|C^N|}$, $X^F \in \mathbb{R}^{T\times|C^F|}$, and $X^P \in \mathbb{R}^{T\times|C^P|}$ denote the native load values in T time steps in buildings of C^N, C^F and C^P, respectively. Similarly, S^N, $S^F \in \mathbb{R}^{T\times|C^F|}$ and $S^P \in \mathbb{R}^{T\times|C^P|}$ represent the PV generations of these buildings. In our setup, X^N , X^F, and S^F are known. S^N is all zero. X^P and S^P are unknown. Only the net load values $N^P = X^P - S^P$ are observed for C^P buildings.

Real-time measurements: Let Z^N in $\mathbb{R}^{2K\times|C^N|}$ denote the real-time native load measurements in C^N buildings. Let Z^F and V^F in $\mathbb{R}^{2K\times|C^F|}$ denote the native load and solar measurements in C^F buildings, and E^P in $\mathbb{R}^{2K\times|C^P|}$ denote the net load measurements in C^P buildings from time $s-2K+1$ to s.

*The **objective** of the native load forecasting problem is to develop a method using historical data X^N, X^F, S^F and N^P such that, when given the input of real-time measurements Z^N, Z^F, V^F and E^P, the method estimates the native load consumption in all buildings at time $t+1$, denoted by $\hat{y}^{s+1}$.*

As illustrated in Fig. 1, the overall approach is to learn a unified prediction and disaggregation model offline using the observed historical data X^N, X^F, S^F, and N^P. In real-time operations, the observed data from time $s-2K+1$ to time s, i.e., Z^N, Z^F, V^F, and E^P, are fed into the trained prediction model, which then forecasts the native load of all buildings at time $s+1$. The main technical challenge is that in both the recorded data and real-time measurements, native loads in buildings C^P are not directly observed, which complicates the prediction of native loads in these buildings.

To address the challenge of missing native load measurements in C^P buildings, our technical approach is to explore the spatial correlations in the native load consumption and PV generations in these nearby buildings to enhance the forecasting performance. Typically, native loads in buildings with similar building purposes, social habits, and weather patterns, may exhibit strong spatial correlation[1]. For example, most native load curves of office buildings may start to rise at about 8 am and start to decline after 6 pm during workdays. The power consumption of air conditioners (AC) largely correlates with the weather conditions. The spatial correlation of PV generations results from similar solar radiation and cloud coverage conditions in the same region [40].

To model these spatial correlations, we use the native load graph, represented by the load adjacency matrix $A \in \mathbb{R}^{N\times N}$, to characterize the correlations among native load consumption in all N buildings. The element A_{ij} in A denotes the strength of spatial correlation between building i and building j. Similarly, we also use the solar graph, represented by the solar adjacency matrix $P \in \mathbb{R}^{M\times M}$, to characterize the correlation of solar generations of these M buildings. Note that A and P are data-dependent and thus not fully known, because both native loads and solar generations are not directly measured in buildings in C^P. Furthermore, A can be time-varying when the selected time interval changes, because the load correlation patterns can change over time. Our major technical contribution is to learn these unknown and time-varying spatial correlations to enhance the native load forecasting accuracy of C^P buildings.

III. The Proposed ULoFo Method

ULoFo uses recorded history data to train a prediction model offline. In online operations, ULoFo sends the real-time measurements as the input to the prediction model to forecast the native load at the next time step. Section III-A summarizes the major component of the offline training methods, and Sections III-B to III-D introduce each component respectively. Section III-E discusses the real-time forecasting implementation of ULoFo. Section III-F analyzes the time complexity of ULoFo.

A. Overview of Offline Training Method

The framework of the training process is shown in Fig.2. In offline training, ULoFo first proposes a graph smoothing disaggregation model (GSD) to disaggregate the load component and solar component in C^P buildings from the recorded net load data. It then employs the prediction model STAWnet [39], which takes the observed and disaggregated load as input and predicts the future load of all buildings. The advantage of our proposed ULoFo method is to jointly and iteratively solve disaggregation and forecasting as a unified process. This is mainly achieved by the proposed two interaction strategies,

[1]The spatial correlation of native loads among commercial buildings is verified by Pearson Correlation defined in (1) using a dataset in a Typical Meteorological Year (TMY3) in California and Texas [19]. The spatial correlation of residential loads has been verified in [14].

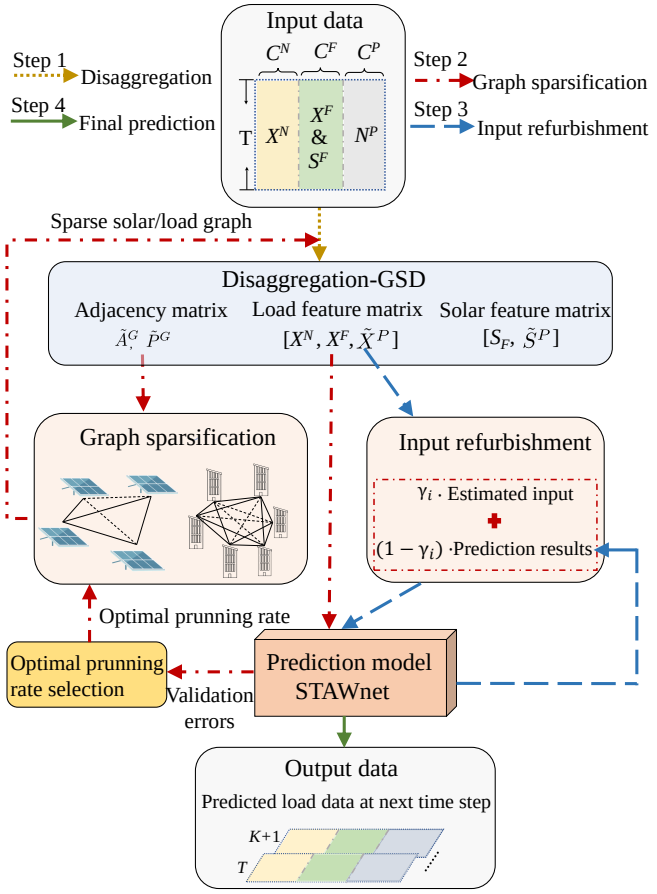

Fig. 2. Overall framework for the offline training of ULoFo

graph sparsification, and input refurbishment, which leverage the forecasting result to enhance the disaggregation performance and prediction performance. The major components are summarized as follows.

1) Disaggregation: The disaggregation model GSD uses a graph smoothing technique by minimizing the Dirichlet energy function over the graphs to estimate native load $\tilde{X}^P$ and PV generation $\tilde{S}^P$ of buildings in C^P, where the load and solar graphs are represented by the current estimated average load adjacency matrix $\tilde{A}^G$ and solar adjacency matrix $\tilde{P}^G$. GSD then updates $\tilde{A}^G$ and $\tilde{P}^G$ by (1) using the newly estimated $\tilde{X}^P$ and $\tilde{S}^P$.

2) Load Prediction: The native load observations in C^N and C^F buildings and the current native load estimation in C^P buildings in a time window of K are fed into the prediction model STAWnet. STAWnet is trained to extract the time-varying local spatial correlation and temporal correlation over the input data and establish the mapping function to predict the native load for every building in the next time step.

3) Graph Sparsification: The two adjacency matrices $\tilde{A}^G$ and $\tilde{P}^G$ used in GSD are pruned to remove redundant information so as to enhance the disaggregation performance. Given a pruning rate r, the resulting pruned $\tilde{A}^G_r$ and $\tilde{P}^G_r$ are fed to GSD, the output of which is then fed to STAWnet for forecasting. The best pruning rate r^* is selected based on the forecasting accuracy.

4) Input refurbishment: Note that the native load consumption in C^P buildings from time $K+1$ to T can be both estimated by the GSD disaggregation model and forecasted by the STAWnet forecasting model. Because the STAWnet model learns the nonlinear spatial correlations among buildings, while GSD mainly focuses on the linear correlations, the forecasting result may enhance the disaggregation result. Input refurbishment enhances the estimation accuracy of the native load in C^P buildings by replacing the current estimation by the weighted sum (with coefficient γ_i) of the current estimation and the corresponding forecasting result by the current forecasting model $f(\cdot, \Theta_i)$, and then sending the updated estimation to load prediction component to train the $i+1$-th forecasting STAWnet model $f(\cdot, \Theta_{i+1})$, where $i = 1, \ldots, I$ and I is the total number of refurbishment times. $f(\cdot, \Theta_{I+1})$ is the final prediction model.

The offline training returns the following: the optimal pruned solar adjacency matrix $P^G_{r^*}$, the estimated solar amplitude $\tilde{\beta}^s$, the weight coefficient γ_i, the learned prediction model $f(\cdot, \Theta_i)$ for each i of total I refurbishment iterations, and the final prediction model $f(\cdot, \Theta_{I+1})$.

B. Graph Smoothing Based Disaggregation Model (GSD)

GSD aims to disaggregate the unknown native loads and PV generations separately from the observed net load of buildings in C^P. Other inputs include the native load and PV measurements from buildings in C^N and C^F. The main idea of GSD, illustrated in Fig. 3, is to explore the spatial correlations among native loads and PV generations, respectively, and estimate the native loads and PV generations in C^P building by aggregating the corresponding features of other buildings.

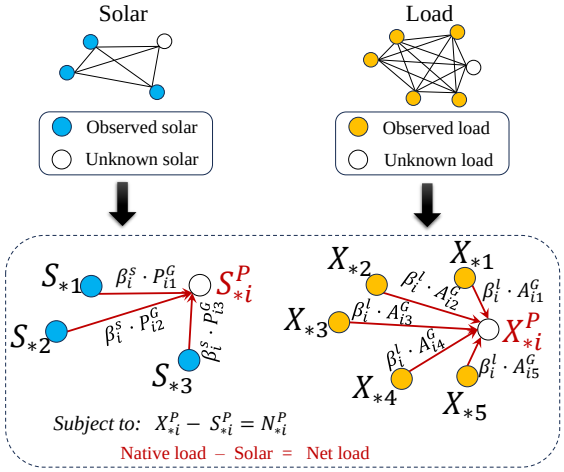

Fig. 3. Illustration of GSD. 1 building in C^P, 3 buildings in C^F, and 2 buildings in C^N. A^G, P^G are computed from (1), and β^l, β^s from (4).

The proposed GSD has three major steps: (i) estimating the patterns of load consumption and solar generation in C^P buildings using the estimated average graph adjacency matrices, (ii) amplitude estimation of the estimated patterns, and (iii) average graph adjacency matrix update using the estimated load consumption and solar generation. We next describe them in detail.

1) Pattern Recovery: The main idea of pattern recovery is based on the assumption of feature homophily [41], meaning

that the spatially correlated nodes, which correspond to buildings here, should have similar features. Here, the characteristic of feature homophily is quantified by the Dirichlet energy function [41], which is a smoothness criterion for the features over a graph. The patterns of the unobserved load and PV data are recovered by minimizing the Dirichlet energy function.

The load correlation graph is represented by the average load adjacency matrix of all T instants in the recorded data, denoted by $A^G \in \mathbb{R}^{N\times N}$. Its element A^G_{ij} denotes the strength of average spatial correlation between buildings i and j.

$$A^G_{ij} = \frac{\text{cov}(X_{*i}, X_{*j})}{\sqrt{\text{cov}(X_{*i}, X_{*i})\text{cov}(X_{*j}, X_{*j})}} \quad i, j = 1, \dots N, \quad (1)$$

where X_{*i} and X_{*j} is the ith column and jth column of X. $\text{cov}(\cdot,\cdot)$ means covariance calculation operator. A^G cannot be fully computed initially because X^P is unknown. The unknown entries in A^G are initialized as random values. The solar adjacency matrix $P^G \in \mathbb{R}^{M\times M}$ is defined in a similar way to represent the average spatial correlation of solar generation in buildings with solar panels.

We aim to recover the load patterns in C^P buildings by minimizing the Dirichlet energy function $l(X, A^G)$ as

$$l(X, A^G) = \frac{1}{2}\text{trace}(X\Delta(X)^{\mathrm{T}}), \quad (2)$$

where $\Delta = I - D^{-\frac{1}{2}}A^G D^{-\frac{1}{2}}$ is graph Laplacian matrix, and the degree matrix D is a diagonal matrix with $D_{ii} = \sum_j A^G_{ij}$. The minimization problem

$$\min_{X^P} \quad l(X, A^G) \quad (3)$$

is solved by the iterative scheme in [41] with X^P initialized to be all zero. Let $\tilde{X}^{\text{pattern\_}P}$ denote the returned solution. $\tilde{X}^{\text{pattern\_}P}$ is used as the estimated native load patterns in C^P buildings because minimizing the Dirichlet energy function returns the load patterns but does not provide an accurate estimation of the load amplitude.

Let $\tilde{S}^{\text{pattern\_}P}$ denote the estimated solar patterns in C^P buildings. The computation of $\tilde{S}^{\text{pattern\_}P}$ has a similar process as above and is skipped.

2) Amplitude Estimation: After estimating the load patterns $\tilde{X}^{\text{pattern\_}P}$ and PV patterns $\tilde{S}^{\text{pattern\_}P}$ of buildings in C^P, we estimate the amplitudes of load and solar, represented by β^l and β^s in $\mathbb{R}^{|C^P|}$, respectively. Specifically, for each building i in C^P, we solve the following problem,

$$\begin{aligned}\min_{\beta^l_i,\beta^s_i} \quad & \lambda\|(\beta^l_i\tilde{X}^{\text{pattern\_}P}_{*i} - N^P_{*i})_\Omega\|^2_F \\ & + \|\beta^l_i\tilde{X}^{\text{pattern\_}P}_{*i} - \beta^s_i\tilde{S}^{\text{pattern\_}P}_{*i} - N^P_{*i}\|^2_F,\end{aligned} \quad (4)$$

where the operator $(\cdot)_\Omega$ sets the values not in set Ω to zero while maintaining the values in set Ω. Ω includes the time instants from 9 pm to 5 am here. The first term of the objective in (4) means that the estimated native load is close to the net load at night. The idea is that, during these time periods, the solar generalization is zero, and the net load only contains the native load. The second term means that the estimated native load minus the estimated solar should be close to the net load at any time. λ is a weight in $(0, 1)$.

Let $\tilde{\beta}^l$ and $\tilde{\beta}^s$ be the estimated amplitude. We obtain the estimated native load values and PV generation from (5):

$$\begin{cases}\tilde{S}^P = \tilde{\beta}^s\tilde{S}^{\text{pattern\_}P} \\ \tilde{X}^P = N^P + \tilde{S}^P.\end{cases} \quad (5)$$

The reason for calculating $\tilde{X}^P$ using $N^P + \tilde{S}^P$, instead of using $\tilde{\beta}^l\tilde{X}^{\text{pattern\_}P}$, is that the PV generation patterns in nearby buildings are generally similar, while the load patterns are more diverse across buildings. Thus, the solar estimation is more accurate, and we estimate the load by adding the net load with solar generalization.

3) Adjacency Matrix Update: After obtaining $\tilde{X}^P$ and $\tilde{S}^P$, we calculate the estimated average load adjacency matrix $\tilde{A}^G$ in (1) and, similarly, the estimated average solar adjacency matrix $\tilde{P}^G$ using $\tilde{X} = [X^N, X^F, \tilde{X}^P]$ and $\tilde{S} = [S^F, \tilde{S}^P]$, respectively.

C. STAWnet Forecasting Model

The load forecasting model aims to learn a function $f(\cdot;\Theta)$, parameterized by trainable weights Θ, that maps the past K steps historical native loads to native load values at the next time step. In this paper, it utilizes the recorded and estimated native load data as input data to construct training samples. After the function $f(\cdot;\Theta)$ is trained, it can be leveraged to forecast future native loads given real-time native load measurements.

Mathematically, let X^N_{t*} denote the tth row of X^N. Let matrices $X^N_t = [X^N_{(t-K+1)*}; \dots; X^N_{t*}]$ and $X^F_t = [X^F_{(t-K+1)*}; \dots; X^F_{t*}]$ denote the observed native load consumptions from time $t-K+1$ to t in C^N and C^F buildings in recorded data, respectively. Let $\tilde{X}^P_t = [\tilde{X}^P_{(t-K+1)*}; \dots; \tilde{X}^P_{t*}]$ denote the estimated native load in C^P buildings from time $t-K+1$ to t. Let $\hat{y}^{t+1}$ in $\mathbb{R}^{1\times N}$ denote the predicted native load values at time $t+1$ of all buildings by the STAWnet model. Then the prediction is

$$[X^N_t, X^F_t, \tilde{X}^P_t] \xrightarrow{f(\cdot;\Theta)} \hat{y}^{t+1}. \quad (6)$$

The learning problem aims to find Θ that minimizes the average mean absolute errors (MAE) between the predicted and actual values of all $T-K$ predictions (T time steps excluding the first K time steps in the recorded data),

$$\begin{aligned}&\min_\Theta \quad \ell_\Theta \\ \text{where} \quad & \ell_\Theta = \tfrac{1}{N(T-K)}\sum_{t=K}^{T-1}\Big(\sum_{i\in C^N}|X^N_{(t+1),i} - \hat{y}^{t+1}_i| + \\ & \sum_{i\in C^F}|X^F_{(t+1),i} - \hat{y}^{t+1}_i| + \sum_{i\in C^P}|\tilde{X}^P_{(t+1),i} - \hat{y}^{t+1}_i|\Big).\end{aligned} \quad (7)$$

Note that because native load X^P is not directly observed, the estimated value $\tilde{X}^P$ by GSD is used in (7) to compute the forecasting error and learn the function $f(:,\Theta)$. The MAE loss function, instead of the widely used ℓ_2 loss function, is adopted in this paper because the MAE loss function is less sensitive to noisy input and outliers.

This paper leverages the STAWnet [39] model to learn the function $f(\cdot;\Theta)$, and the main architecture of STAWnet is shown in Fig. 4. The input data are first transformed by a

convolutional layer (Conv) and then fed into four spatial-temporal blocks (ST-block). Every ST-block contains a residual connection and is skip-connected to the output layer. The two most important parts of STAWnet are the gated temporal convolution network (Gated TCN) and dynamic attention network (DAN), which extract time-varying temporal correlations and spatial correlations, respectively. Compared

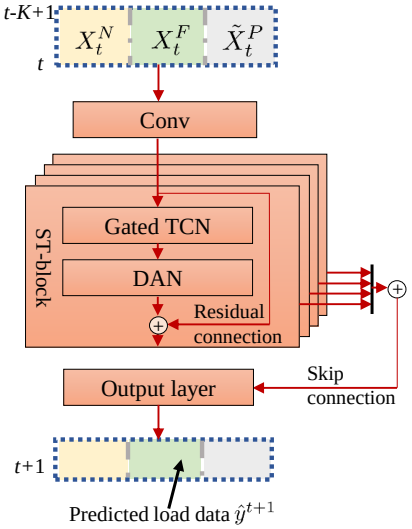

Fig. 4. Prediction model by STAWnet [39]

with most existing forecasting models that are applied in load prediction, STAWnet has a distinct ability to capture the time-varying spatial correlations among native load values through the DAN network. Hence, we briefly introduce DAN here, and more details can be found in [39].

The key idea of DAN is to dynamically assign different weights to different graph nodes, which correspond to buildings, when aggregating the information of neighboring nodes. Let A_{ij}^t denote the correlation of nodes i and j at time t, node i aggregates the information of all nodes by

$$\mathbf{x}'_{ti} = \sum_{j=1}^{N} A_{ij}^t \cdot \mathbf{x}_{tj} \quad \text{s.t.} \sum_{j=1}^{N} A_{ij}^t = 1, \tag{8}$$

where $\mathbf{x}_{tj}$ is the hidden state of node j for some layer, and $\mathbf{x}'_{ti}$ is the hidden state of node i in the next layer.

To better compute the dynamic correlation A_{ij}^t, a node embedding is learned for every node in DAN. Let e_i and e_j denote the node embedding of the node i and the node j, respectively. A_{ij}^t can be computed by

$$o_{ij}^t = \frac{\langle W_q(\mathbf{x}_{ti} \| e_i), W_k(\mathbf{x}_{tj} \| e_j) \rangle}{\sqrt{d_1}} \tag{9}$$

$$A_{ij}^t = \frac{\exp(o_{ij}^t)}{\sum_{k=1}^{N} \exp(o_{ik}^t)} \tag{10}$$

where $\|$ and $\langle \cdot, \cdot \rangle$ denote the concatenation and inner product operations, respectively. d_1 denotes the dimension of e_i and e_j. W_q and W_k are the trainable query and key matrices, respectively.

Note that STAWnet will be trained again every time the input is refurbished. Let $f(\cdot, \Theta_i)$ $(i = 1, ..., I+1)$ denote all $I+1$ trained models where I is the total number of input refurbishment.

D. Interaction of Disaggregation and Forecasting

We propose two interaction strategies, graph sparsification, and input refurbishment, to leverage the prediction result to improve the disaggregation performance, which in turn further enhances the prediction accuracy.

1) Graph Sparsification: Graph sparsification removes the redundant edges in a graph and can enhance accuracy. Specifically, we sparsify the estimated average load and solar adjacency matrices $\tilde{A}^G$ and $\tilde{P}^G$ by keeping the top $1-r$ fraction of the largest entries in each column and setting all other r fraction entries to zero, where r is viewed as the pruning rate. Intuitively, as r increases, the disaggregation performance will first improve due to the removal of redundant information. If r becomes too large, the disaggregation performance will then decrease because important correlation information may also be removed. Due to the lack of ground-truth native load consumption in C^P buildings, one cannot directly compute the best value r^* to optimize the disaggregation performance. Therefore, we propose to leverage the prediction result to guide the selection of r^* in this paper. Because accurate disaggregated load values can lead to small predicted errors, we employ the prediction result to select the optimal r^*. Specifically, we prune $\tilde{A}^G$ and $\tilde{P}^G$ with different r and use the resulting matrices in GSD for load disaggregation. The disaggregated values are then sent to STAWnet for load forecasting. We pick the r value that corresponds to the smallest validation error of the forecasting results as the optimal r^*. Let $\tilde{A}_{r^*}^G$ and $\tilde{P}_{r^*}^G$ denote the resulting optimally pruned adjacency matrices.

2) Input Refurbishment: Note that the native load consumption in C^P buildings from time $K+1$ to T is first estimated by the GSD disaggregation model and then forecasted by the STAWnet forecasting model. The idea of input refurbishment is to use the weighted sum of disaggregated estimation and the forecasting estimation as the new input to the STAWnet model and run the forecasting model again. The refurbishment process can be repeated on the new input obtained from the previous refurbishment. As shown in Fig. 5, we compute the weighted sum of the native load estimation $\tilde{X}^P$ and its corresponding prediction result $\hat{y}$, using γ as the weight. The result is then treated as the updated $\tilde{X}^P$. The processes of conducting weighted sum of $\tilde{X}^P$ and $\hat{y}$, and then updating $\tilde{X}^P$, are repeated total I times.

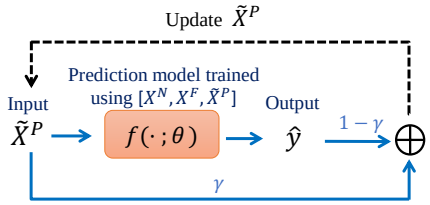

Fig. 5. Illustration of input refurbishment (γ is computed by the statistic characteristics of loss information obtained from prediction model $f(\cdot;\theta)$).

Specifically, let Ω_d be a subset of $[K+1, T]$ that contains all the time indices corresponding to the daytime from 10 am

to 5 pm. We update the estimated native load $\tilde{X}_{ti}^P$ through input refurbishment by

$$\tilde{X}_{ti}^P \leftarrow \gamma \tilde{X}_{ti}^P + (1-\gamma)\hat{y}_i^t, \quad \forall t \in \Omega_d, i \in C^P \tag{11}$$

and no refurbishment for other time t and building i. On the righthand side (RHS) of (11), the initial $\tilde{X}_{ti}^P$ is obtained from (5) in disaggregation, and $\hat{y}_i^t$ is its forecasting result from (7). The observed net load at night time is mostly native load because the solar generation is negligible. Thus, we do not refurbish the input during the nighttime. After the new estimation $\tilde{X}^P$ is obtained, the estimated solar is updated by

$$\tilde{S}^P = \tilde{X}^P - N^P. \tag{12}$$

Then the new $\tilde{X}^P$ is used to train the STAWnet model again.

γ in $(0, 1)$ is the constant weight factor to be determined. Intuitively, if the load disaggregation result $\tilde{X}_{ti}^P$ is accurate, γ should be large. Otherwise, γ should be small. However, because the ground truth native load is unknown for C^P buildings, one cannot directly determine whether $\tilde{X}_{ti}^P$ is accurate or not. Here, we adopt and improve the method in [42] that computes γ for label refurbishment of noisy labels in supervised learning.

The idea is to estimate the probability of every $\tilde{X}_{ti}^P$ being accurate based on its corresponding forecasting loss $|\tilde{X}_{ti}^P - \hat{y}_i^t|$ for every t in Ω_d and i in C^P, and then use the average probability of all these samples $\tilde{X}_{ti}^P$ being accurate as γ. Compared with accurate samples, inaccurate samples yield larger loss values and less consistent predictions during initial training [43], which means the forecasting losses of inaccurate samples tend to have larger mean values and variance values. Therefore, the forecasting losses of all these samples $\tilde{X}_{ti}^P$ can be divided into two groups with different distributions. Determining whether $\tilde{X}_{ti}^P$ is accurate or not is equivalent to determining which group its corresponding loss value $|\tilde{X}_{ti}^P - \hat{y}_i^t|$ belongs to.

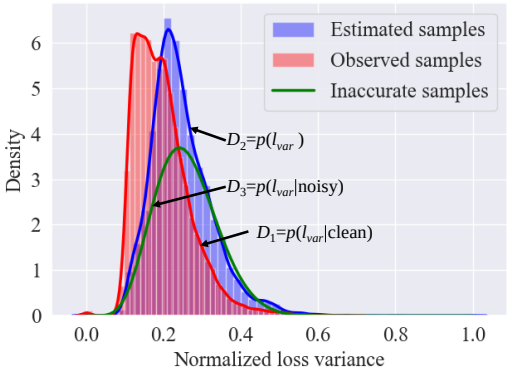

Fig. 6. Loss variance distribution of observed samples and estimated samples ($D_2 = 0.25D_1 + 0.75D_3$)

The phenomenon of two different distributions has been observed in [43]; we also demonstrate that on the forecasting loss variances in Fig. 6. D_2 shows the loss variance distribution of prediction values in C^P buildings. D_2 can be numerically decomposed as the weighted sum of two distributions D_1 and D_3. D_1 is the loss variance distribution of prediction values in C^N and C^F buildings. Because native load values are directly measured in these buildings, the forecasting values of these samples are considered accurate. The calculated distribution based on the decomposition D_3 is treated as the loss variance distribution of inaccurate estimations.

To compute γ mathematically, we first fit the probability distribution of loss variance, denoted by l_{var}, using a two-component Beta mixture model (BMM) [42]. The group of samples with accurate disaggregation results is referred to as the "clean" group, denoted by the case of $k = 1$, and the other group of inaccurate disaggregation results are referred to as the "noisy" group, denoted by the case of $k = 2$. The overall distribution of the loss variance can be represented as

$$p(l_{\text{var}}) = \sum_{k=1}^{2} p(k)p(l_{\text{var}}|k) \quad s.t. \sum_{k=1}^{2} p(k) = 1, \tag{13}$$

where $p(k = 1)$ and $p(k = 2)$ are the prior probability that the disaggregation results are accurate and inaccurate, respectively. $p(l_{\text{var}}|k)$ is the conditional probability of l_{var} when the disaggregation is accurate or inaccurate, which follows a beta distribution

$$p(l_{\text{var}}|k) = \frac{\Gamma(\alpha_k + \beta_k)}{\Gamma(\alpha_k)\Gamma(\beta_k)} l_{\text{var}}^{\alpha_k - 1}(1 - l_{\text{var}})^{\beta_k - 1}, \tag{14}$$

where $\alpha_k, \beta_k > 0$ are the variables that need to be estimated to determine the distribution $p(l_{\text{var}}|k)$. $\Gamma(\cdot)$ is the Gamma function. The prior and conditional probabilities in the RHS of (13) can be estimated using Expectation Maximization (EM) algorithms [44].

After estimating each term in the RHS of (13) and computing (13), for any t in Ω_d and any building i in C^P, the conditional probability that the corresponding disaggregation result $\tilde{X}_{ti}^P$ is accurate given the observed forecasting loss variance $l_{\text{var\_}ti}$ can be computed by

$$p(k = 1|l_{\text{var}} = l_{\text{var\_}ti}) = \frac{p(k = 1)p(l_{\text{var}} = l_{\text{var\_}ti}|k = 1)}{p(l_{\text{var}} = l_{\text{var\_}ti})}. \tag{15}$$

The parameter γ is computed by

$$\gamma = |\Omega_d|^{-1}|C^P|^{-1} \sum_{t \in \Omega_d, i \in C^P} p(k = 1|l_{\text{var}} = l_{\text{var\_}ti}). \tag{16}$$

Our method of computing γ improves over that in [42] in two aspects. First, we employ the loss variance across all training epochs rather than one loss value in estimating the two-component model. That is because using the loss values in only one epoch may lead to unstable statistical distributions. Second, instead of estimating all four terms in the RHS of (13) together, we first estimate $p(l_{\text{var}}|k = 1)$ using the forecasting loss variances in C^F and C^N buildings, because native load values in these buildings are directly measured and thus accurate. Then, we fix $p(l_{\text{var}}|k = 1)$ and estimate the other three terms in the RHS of (13) using the EM algorithm. Our approach is more accurate and computationally efficient than estimating all four terms together in [42]. The calculation details of γ based on the improved BMM are shown in section A in the supplementary material.

E. Real-time Load Forecasting

During the real-time test stage of ULoFo, we will leverage the following quantities leaned off-line: the pruned solar adjacency matrix with optimal pruning rate $\tilde{P}^G_{r^*}$, the estimated solar amplitude $\tilde{\beta}^s$, γ_i in the ith input refurbishment, and the ith prediction model $f(\cdot, \Theta_i)$ for $i = 1, ..., I$, and the final prediction model $f(\cdot, \Theta_{I+1})$. At time s, we will use the observed data from time $s - 2K + 1$ to time s to predict the native load at time $s + 1$ in all N buildings. Fig. 7 illustrates the whole real-time test implementation of ULoFo.

Mathematically, let Z^N in $\mathbb{R}^{2K\times|C^N|}$ denote the native load measurement in C^N buildings, Z^F and V^F in $\mathbb{R}^{2K\times|C^F|}$ denote the native load and solar measurements in C^F buildings, and E^P in $\mathbb{R}^{2K\times|C^P|}$ denote the net load measurements in C^P buildings from time $s - 2K + 1$ to s. Because the native load patterns can be diverse across buildings, while the solar patterns across buildings share more similarities, we first estimate the solar generation $\tilde{V}^P$ and then combine it with the net load observations E^P to calculate the native load estimation $\tilde{Z}^P$.

Specifically, first, the solar pattern is recovered by minimizing the Dirichlet energy function

$$\min_{V^{\text{pattern\_}P}} l(V', P^G_{r^*}), \tag{17}$$

where $V' = [V^F, V^{\text{pattern\_}P}]$ and let $\tilde{V}^{\text{pattern\_}P}$ denote the estimated solar pattern. The solar and load estimations can be calculated by

$$\begin{cases} \tilde{V}^P = \tilde{\beta}^s \tilde{V}^{\text{pattern\_}P} \\ \tilde{Z}^P = E^P + \tilde{V}^P, \end{cases} \tag{18}$$

where the magnitude $\tilde{\beta}^s$ is learned from the off-line training.

Then for every iteration i from 1 to I, the prediction model $f(\cdot, \Theta_i)$ uses the data in $[Z^N, Z^F, \tilde{Z}^P]$ in every time window of length K to predict the native load in all buildings for the next time. Let Q^i in $\mathbb{R}^{K\times|C^P|}$ denote the forecasting result by $f(\cdot, \Theta_i)$ for the native load in C^P buildings from time $s - K + 1$ to s. We apply the input refurbishment to $\tilde{Z}^P$ from time $s - K + 1$ to s, i.e.,

$$\tilde{Z}^P_{k*} \leftarrow \gamma_i \tilde{Z}^P_{k*} + (1 - \gamma_i) Q^i_{k*} \quad \forall k \in [s - K + 1, s]. \tag{19}$$

Finally, the prediction of native load at $s + 1$ is obtained by sending the updated $[Z^N, Z^F, \tilde{Z}^P]$ to the final prediction model $f(\cdot, \Theta_{I+1})$.

F. Time Complexity Analysis

The computational complexity of ULoFo is dominated by the computation needed by GSD and STAWnet, while the computation of graph sparsification and input refurbishment is relatively small. Thus, we focus on analyzing the time complexity of GSD and STAWnet.

Offline Training. In the step of pattern recovery in GSD, the time complexity of solving (3) is $\mathcal{O}(N^2T)$ according to the solving scheme in [41]. The time complexity of solving (4) in the step of amplitude estimation in GSD is $\mathcal{O}(NT)$. Thus, the time complexity of GSD in the offline training stage is $\mathcal{O}(N^2T)$.

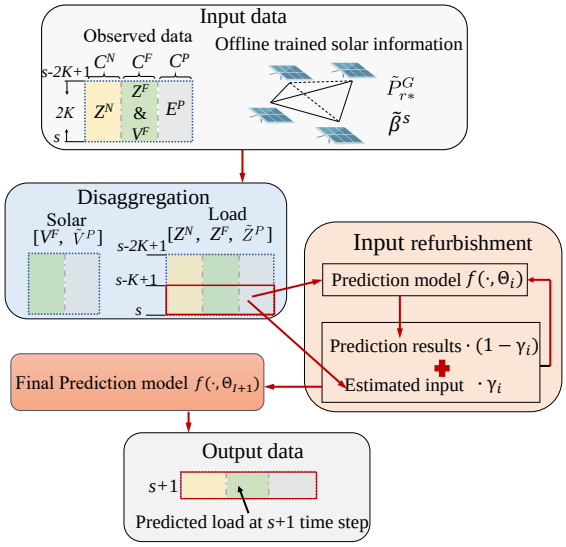

Fig. 7. Real-time implementation of ULoFo

The main modules of STAWnet contain the input convolutional layer, ST-block, residual connection, skip connection, and output layer. Except for ST-block, the remaining modules only involve convolution computation with kernel size 1×1 and non-linear activation by rectified linear unit (ReLU). Give a training sample $\mathbf{s} \in \mathbb{R}^{d_{in}\times N\times K}$ through a convolutional layer with kernel size 1×1, the time complexity is $\mathcal{O}(NKd_{in}d_{out})$, where d_{in} and d_{out} represent the feature dimension of sample $\mathbf{s}$ before and after convolution, respectively. Note that d_{in} and d_{out} are fixed feature dimensions and can be viewed as constants. Thus, the time complexity of sample $\mathbf{s}$ in the modules, including the input layer, the output layer, residual connection, and skip connection, is $\mathcal{O}(NK)$. One ST-block includes Gated TCN and DAN. Since Gated TCN applies convolutional operation with kernel size 1×1, the time complexity of sample $\mathbf{s}$ through Gated TCN is $\mathcal{O}(NK)$. In the module DAN, the dominant time complexity of (8) is $\mathcal{O}(N^2d_1K)$, and the time complexity of (9) is $\mathcal{O}(N^2)$. Since d_1 is a small constant, the time complexity of the DAN module is $\mathcal{O}(N^2K)$. Therefore, the time complexity of STAWnet is $\mathcal{O}((T-K)N^2K)$, where $T-K$ represents the total training samples, respectively. Thus, the time complexity of ULoFo in the offline stage is $\mathcal{O}((T-K)N^2K + N^2T)$.

Real-time forecasting. The time complexity of solving (17) is $\mathcal{O}(M^2K)$. The inference of STAWnet for one testing sample can be computed by $\mathcal{O}(N^2K)$. Thus, the time complexity of ULoFo in the real-time test stage is $\mathcal{O}(N^2K + M^2K)$.

IV. Numerical Experiments

Our method is evaluated on one commercial building dataset and one residential building dataset. Due to the space limit, we only present the results on the commercial buildings here, and the results on residential buildings are in the supplementary material.

A. Dataset Description

The native load values of 40 commercial buildings are also selected from a Typical Meteorological Year (TMY3) dataset

in Texas [19][2], in which 20 buildings are small offices and 20 buildings are quick-service restaurants. We randomly select 10 buildings, 15 buildings, and 15 buildings from the 40 commercial buildings to construct the commercial building set C^N, C^F, and C^P, respectively. The PV generations of 30 sites in Texas for the experiment on commercial buildings are randomly chosen from the public dataset in [45], [2]. According to the maximum capacity of solar and peak load values of DOE commercial reference buildings in [46], we scale the selected native load values and PV generations. The scaled PV generations are added to the scaled native loads of commercial buildings in C^F and C^P to obtain the net loads. The number of commercial buildings in C^N, C^F, and C^P are 10, 15, and 15, respectively. Note that the maximum capacity of PV output of buildings in C^P is less than 10kWp since the BTM PV usually refers to small-scale PV systems (<10kWp) [47].

Both the commercial load profile and PV generation profile have 8760 time steps with one-hour resolution in one year. The one-year load data are divided into training, validation, and test datasets by the ratio of 0.7: 0.2: 0.1.

B. Methods and Evaluation Metric

Because no existing method solves the disaggregation and native load prediction jointly without requiring historical data for BTM consumers, this paper compares three disaggregation baselines and four prediction baselines for the disaggregation and prediction performance separately. Then, the ULoFo method is compared with the combinations of these methods. Three disaggregation baselines include the Consumer Mixture Model (CMM) [35], Discitionary Learning and Sparse Disaggregation (DL-SD) [38], and the Two-layer Approach (TLA) [4]. CMM constructs two alternative optimization steps by using dominant load and solar consumption patterns. DL-SD is a technique that learns the optimal dictionary and sparse coefficients for disaggregating loads and PV generations. TLA utilizes the spatial correlations inherent in native demands and PV generations to accomplish disaggregation. Four prediction baselines include GraphWavenet [14], Spatio-Temporal Graph Convolutional Networks (STGCN) [48], Multi-Scale Adaptive Graph Neural Networks (MAGNN) [49], and Non-stationary Transformers (NST) [50]. GraphWavenet includes eight spatial-temporal layers. STGCN contains two spatial-temporal blocks. MAGNN is a multi-scale adaptive graph neural network. NTS is a deep and complex non-stationary transformer model.

To evaluate the joint disaggregation and forecasting performance, we compare our proposed ULoFo method with two combinations of the disaggregation method CMM and two prediction models, GraphWavenet and STGCN, respectively. They are abbreviated by CMM-GW and CMM-STGCN, respectively. We only keep these two combinations because they perform the best among all these combinations of three disaggregation and four forecasting baselines. We use the K-Means technique instead of the K-Shape method when implementing CMM to construct the load patterns since the K-Shape method performs poorly and slowly in our dataset. We use the observed PV profiles to construct PV patterns, instead of simulated PV patterns in [4] when implementing TLA. That is because PV data is observed at some buildings in our setup. The hyperparameter settings of all methods are in the supplementary material.

Both disaggregation and prediction performance are evaluated by mean absolute percentage errors (MAPE), mean absolute errors (MAE), and root mean square errors (RMSE). For example, the load MAPE, load MAE, and load RMSE for one building i is calculated by

$$\begin{aligned} \text{Load MAPE}_i &= \frac{\sum_{s\in\Omega_{\text{test}}} |\tilde{E}_{si} - E_{si}|}{|\sum_{s\in\Omega_{\text{test}}} E_{si}|} \times 100\% \\ \text{Load MAE}_i &= \sum_{s\in\Omega_{\text{test}}} |\tilde{E}_{si} - E_{si}| \\ \text{Load RMSE}_i &= \sqrt{\frac{\sum_{s\in\Omega_{\text{test}}} (\tilde{E}_{si} - E_{si})^2}{|\Omega_{\text{test}}|}}, \end{aligned} \tag{20}$$

where Ω_{test} denotes the set of time indices of testing samples. E_{si} and $\tilde{E}_{si}$ denote the ground-truth values and the disaggregation (or forecasting) value of building i at time s.

The average Load MAPE for all buildings is

$$\begin{aligned} \text{Load MAPE} &= \frac{\sum_{i=1}^{N} \text{Load MAPE}_i}{N} \\ \text{Load MAE} &= \frac{\sum_{i=1}^{N} \text{Load MAE}_i}{N} \\ \text{Load RMSE} &= \frac{\sum_{i=1}^{N} \text{Load RMSE}_i}{N} \end{aligned} \tag{21}$$

The evaluation metrics for solar can be calculated similarly.

C. Experiment on Commercial buildings

1) Performance Evaluation and Comparison: As shown in Table I, the proposed ULoFo achieves the smallest disaggregation and prediction errors among the three methods for commercial load forecasting. ULoFo has the highest offline computational cost. The real-time process of ULoFo takes 14.3s, which is almost twice the time of the other methods. That is because the input refurbishment in ULoFo is performed three times here, which results in more inference time. The inference time can be reduced by reducing the number of implementations of input refurbishment. In fact, as shown later in Table V, without the input refurbishment component, ULoFo takes 8.31s for inference, which is slightly only higher than the inference time of the baselines. In addition, ULoFo still outperforms CMM-STGCN and CMM-GW without input refurbishment. Because CMM-STGCN and CMM-GW use the same disaggregation model, their disaggregation performance is the same. Fig. 8 shows the predicted native load values of one office building during the time period from 8525h to 8545h. The time period from 8528h to 8542h

[2]The native load datasets are accessible at the following link { https://catalog.data.gov/dataset/commercial-and-residential-hourly-load-profiles-for-all-tmy3-locations-in-the-united-state-bbc75}. We thank the authors of [2] for kindly sharing the PV generation data with us.

[3]The units of MAE and MAPE are kilowatts (kW).

TABLE I
COMPARISONS AMONG ULOFO AND BASELINES ON COMMERCIAL BUILDINGS [3]

Methods	Disaggregation						Prediction			Computation time (s)	
	Load error			Solar error			Test error			Offline	Real time
	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE		
CMM-STGCN	0.207	5.58%	0.497	0.207	14.86%	0.497	0.295	6.26%	0.508	512.5	7.1
CMM-GW	0.207	5.58%	0.497	0.207	14.86%	0.497	0.26	6.02%	0.502	1516.1	6.2
ULoFo (ours)	**0.131**	**3.7%**	**0.358**	**0.131**	**10.12%**	**0.358**	**0.195**	**4.28%**	**0.381**	3961.3	14.3

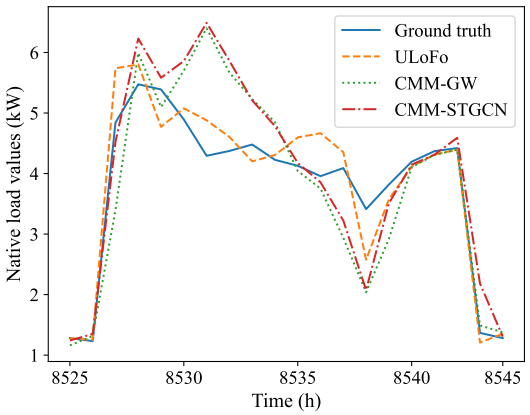

Fig. 8. Prediction results of one office building

TABLE II
LOAD PREDICTION PERFORMANCE ON COMMERCIAL BUILDINGS WHERE THE NATIVE LOAD IS MEASURED AND NOT MEASURED

Methods	Load measured			Load not measured		
	MAE	MAPE	RMSE	MAE	MAPE	RMSE
CMM-STGCN	0.282	4.88%	0.492	0.316	8.57%	0.533
CMM-GW	0.225	4.31%	0.458	0.319	8.86%	0.567
ULoFo (ours)	**0.179**	**3.22%**	**0.348**	**0.222**	**6.05%**	**0.429**

corresponds to 8:00 am to 10:00 pm, during which the load consumption of commercial buildings is high due to human activities. One can observe that the prediction result by ULoFo is the closest to the ground truth data. The predictions by the two baselines around 8530h and 8537h deviate from the ground truth due to the inaccurate disaggregation results. Table II shows the load prediction errors in C^N and C^F where the native load is directly measured, as well as C^P buildings, where the native load is not directly measured. Naturally, all methods perform better when the native load is directly observed. ULoFo performs the best among all three methods in both cases, and the advantage of ULoFo is more significant in buildings where the load is not directly observed.

2) Performance of Disaggregation or Forecasting Only: Here, we compare ULoFo with existing methods for the special cases of disaggregation or forecasting only. For disaggregation only, we only implement the GSD component of ULoFo for training, and the real-time part only disaggregates the data. Table III compares GSD with TLA, CMM, and DL-SD for the disaggregation results and training time. Among the four disaggregation methods, CMM performs the best due to the two alternative optimization steps. The performance of GSD is comparable with the performance of DL-SD. TLA performs the worst because of the large estimated PV peak generation errors, the maximum of which is 62%. Overall, the performance of GSD is comparable to other disaggregation baselines. Moreover, GSD has the lowest computation cost. Note that compared with the disaggregation performance of ULoFo in Table I, the disaggregation error by using the GSD component only is larger because graph sparsification and input refurbishment are removed here.

TABLE III
DISAGGREGATION ONLY PERFORMANCE WITH DIRECTLY MEASURED COMMERCIAL LOAD

Methods	Load error			Solar error			Time (s)
	MAE	MAPE	RMSE	MAE	MAPE	RMSE	
TLA	0.475	12.11%	1.002	0.475	33.6%	1.002	279.6
CMM	**0.207**	**5.58%**	**0.497**	**0.207**	**14.86%**	**0.497**	10.03
DL-SD	0.24	6.75%	0.55	0.24	17.24%	0.55	699.42
GSD	0.261	7.29%	0.628	0.261	20.29%	0.628	**0.38**

TABLE IV
PREDICTION ONLY PERFORMANCE WITH DIRECTLY MEASURED COMMERCIAL LOAD

Methods	Load prediction error			Parameter
	MAE	MAPE	RMSE	
GraphWavenet	0.194	4.14%	0.402	267649
STGCN	0.245	4.76%	0.435	159105
MAGNN	0.295	5.7%	0.579	1303285
NTS	0.48	8.79%	0.796	10856808
STAWnet	**0.15**	**3.18%**	**0.306**	271841

We then compare the prediction only performance of these methods, STAWnet, GraphWavenet, STGCN, MAGNN, and NTS, when the native load is already directly measured. As shown in Table IV, the method, STAWnet, adopted in this paper significantly outperforms the other four methods in real-time commercial building load forecasting. In our setup, the training data is not large enough for both large models, MAGNN and NTS, resulting in poor performance of the two methods. Compared with STGCN, GraphWavenet has more spatial-temporal layers and thus can learn the spatial-temporal correlations better. The forecasting method, STAWnet, adopted in our paper, has an architecture similar to GraphWavenet. The only difference is that STAWnet can additionally learn the dynamic spatial correlation among the input data, leading to improved performance over GraphWavenet. Note that the forecasting accuracy of our method in Table IV is higher than that in Table I. That is because the availability of accurate historical data can enhance forecasting accuracy.

3) Performance of Graph sparsification: Here, we evaluate the graph sparsification strategy. The evaluation metric MAPE

is used to select the optimal pruning rate. We vary the pruning rate r from 0 to 0.9 with a step size of 0.15. Fig. 9 shows how the validation MAPE and test MAPE of load prediction change. One can see that both curves have the same trend of decreasing first for a large range of r until $r = 0.75$ and then increasing for a very large pruning rate. This justifies our strategy of using the validation MAPE as the criterion to select the best r. Fig. 10 also shows the disaggregation MAPE, both of which have the same trend as the forecasting MAPE.

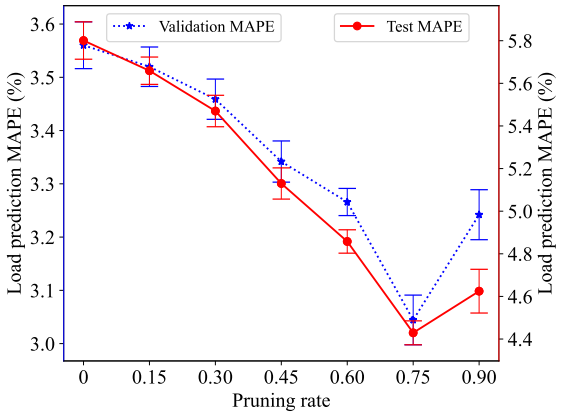

Fig. 9. Commercial prediction performance under different pruning rate

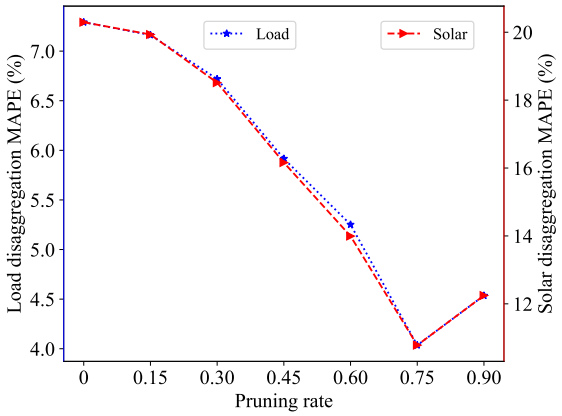

Fig. 10. Commercial disaggregation performance under different pruning rate

4) Performance of input refurbishment: From Table V, one can see the load disaggregation MAPE are improved from 4.03% to 3.7%, and the solar disaggregation MAPE are improved from 10.78% to 10.12% after three times refurbishments. The prediction performance is improved from 4.43% to 4.28%. Clearly, the input refurbishment improves the accuracy of both disaggregation and forecasting.

TABLE V
INPUT REFURBISHMENT ON COMMERCIAL EXPERIMENT

Times	γ	Disaggregation		Prediction	Inference Time (s)
		Load MAPE	Solar MAPE	Load MAPE	
0	-	4.03%	10.78%	4.43%	8.31
1	0.33	3.79%	10.19%	-	-
2	0.41	3.73%	10.13%	-	-
3	0.43	**3.7%**	**10.12%**	**4.28%**	14.3

Here, the total number of refurbished times I is set to 3. A rule to determine I is as follows. Note that the weight parameter γ_i reflects the probability of the currently estimated input being accurate. When γ_i does not change obviously, so does the accuracy of the estimated input, then the refurbished process can stop. TABLE V shows how γ_i and the performance of the estimated input change during the refurbishment process. One can see that the γ increases during the refurbishment process, and so does the accuracy of the estimated result. Moreover, from $i = 2$ to 3, γ_i does not change significantly, and the estimated result has a minor improvement. Thus, the total refurbishment times I is set to be 3.

V. CONCLUSION

With the rapid expansion of PV installations, an increasing number of building-level PV installations are BTM to reduce consumer costs and support the trends towards a decentralized grid. However, the invisible native load measurements resulting from the BTM PV installations make building-level native load forecasting challenging. This paper develops a load forecasting method, referred to as ULoFo, that integrates both load disaggregation and forecasting together for the first time to enhance the forecasting accuracy when the BTM load is not measured. The idea is to explore the unknown and time-varying spatial correlations of multiple nearby buildings. ULoFo contains a load disaggregation module, a load prediction module, and two interaction strategies between them to reduce error propagation and enhance accuracy. ULoFo does not require any historical native load measurements on BTM consumers and demonstrates its superior performance over existing load disaggregation and forecasting methods on practical datasets. Future work includes improving the efficiency of the training process by integrating the disaggregation and prediction in a weakly supervised neural network.

REFERENCES

[1] G. Chitalia, M. Pipattanasomporn, V. Garg, and S. Rahman, "Robust short-term electrical load forecasting framework for commercial buildings using deep recurrent neural networks," *Applied Energy*, vol. 278, p. 115410, 2020.

[2] F. Bu, K. Dehghanpour, Y. Yuan, Z. Wang, and Y. Guo, "Disaggregating customer-level behind-the-meter pv generation using smart meter data and solar exemplars," *IEEE Transactions on Power Systems*, vol. 36, no. 6, pp. 5417–5427, 2021.

[3] M. Khodayar, G. Liu, J. Wang, O. Kaynak, and M. E. Khodayar, "Spatiotemporal behind-the-meter load and pv power forecasting via deep graph dictionary learning," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 10, pp. 4713–4727, 2020.

[4] F. Bu, R. Cheng, and Z. Wang, "A two-layer approach for estimating behind-the-meter pv generation using smart meter data," *IEEE Transactions on Power Systems*, vol. 38, no. 1, pp. 885–896, 2022.

[5] B. Ghaleb and M. Asif, "Application of solar pv in commercial buildings: Utilizability of rooftops," *Energy and Buildings*, vol. 257, p. 111774, 2022.

[6] N. Liu, Q. Chen, J. Liu, X. Lu, P. Li, J. Lei, and J. Zhang, "A heuristic operation strategy for commercial building microgrids containing evs and pv system," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 4, pp. 2560–2570, 2014.

[7] J. Lin, J. Ma, and J. Zhu, "A privacy-preserving federated learning method for probabilistic community-level behind-the-meter solar generation disaggregation," *IEEE Transactions on Smart Grid*, vol. 13, no. 1, pp. 268–279, 2021.

[8] C. C. Liu, H. Chen, J. Shi, and L. Chen, "Self-supervised learning method for consumer-level behind-the-meter pv estimation," *Applied Energy*, vol. 326, p. 119961, 2022.

[9] N. Amral, C. Ozveren, and D. King, "Short term load forecasting using multiple linear regression," in *2007 42nd International universities power engineering conference*. IEEE, 2007, pp. 1192–1198.

[10] W. R. Christiaanse, "Short-term load forecasting using general exponential smoothing," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-90, no. 2, pp. 900–911, 1971.

[11] J.-F. Chen, W.-M. Wang, and C.-M. Huang, "Analysis of an adaptive time-series autoregressive moving-average (arma) model for short-term load forecasting," *Electric Power Systems Research*, vol. 34, no. 3, pp. 187–196, 1995.

[12] Y. Chakhchoukh, P. Panciatici, and L. Mili, "Electric load forecasting based on statistical robust methods," *IEEE Transactions on Power Systems*, vol. 26, no. 3, pp. 982–991, 2010.

[13] M. Q. Raza and A. Khosravi, "A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings," *Renewable and Sustainable Energy Reviews*, vol. 50, pp. 1352–1372, 2015.

[14] W. Lin, D. Wu, and B. Boulet, "Spatial-temporal residential short-term load forecasting via graph neural networks," *IEEE Transactions on Smart Grid*, vol. 12, no. 6, pp. 5373–5384, 2021.

[15] A. Kavousi-Fard, H. Samet, and F. Marzbani, "A new hybrid modified firefly algorithm and support vector regression model for accurate short term load forecasting," *Expert systems with applications*, vol. 41, no. 13, pp. 6047–6056, 2014.

[16] G. Chitalia, M. Pipattanasomporn, V. Garg, and S. Rahman, "Robust short-term electrical load forecasting framework for commercial buildings using deep recurrent neural networks," *Applied Energy*, vol. 278, p. 115410, 2020.

[17] Y. Wang, M. Liu, Z. Bao, and S. Zhang, "Short-term load forecasting with multi-source data using gated recurrent unit neural networks," *Energies*, vol. 11, no. 5, p. 1138, 2018.

[18] N. M. M. Bendaoud and N. Farah, "Using deep learning for short-term load forecasting," *Neural computing and applications*, vol. 32, pp. 15 029–15 041, 2020.

[19] Z. Wu, Y. Mu, S. Deng, and Y. Li, "Spatial–temporal short-term load forecasting framework via k-shape time series clustering method and graph convolutional networks," *Energy Reports*, vol. 8, pp. 8752–8766, 2022.

[20] Z. Kong, C. Zhang, H. Lv, F. Xiong, and Z. Fu, "Multimodal feature extraction and fusion deep neural networks for short-term load forecasting," *IEEE Access*, vol. 8, pp. 185 373–185 383, 2020.

[21] M. Khodayar, J. Wang, and M. Manthouri, "Interval deep generative neural network for wind speed forecasting," *IEEE Transactions on Smart Grid*, vol. 10, no. 4, pp. 3974–3989, 2018.

[22] M. Khodayar, O. Kaynak, and M. E. Khodayar, "Rough deep neural architecture for short-term wind speed forecasting," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 6, pp. 2770–2779, 2017.

[23] 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.

[24] A.-N. Khan, N. Iqbal, A. Rizwan, R. Ahmad, and D.-H. Kim, "An ensemble energy consumption forecasting model based on spatial-temporal clustering analysis in residential buildings," *Energies*, vol. 14, no. 11, p. 3020, 2021.

[25] M. Ganjouri, M. Moattari, A. Forouzantabar, and M. Azadi, "Spatial-temporal learning structure for short-term load forecasting," *IET Generation, Transmission & Distribution*, 2022.

[26] Y. Wang, N. Zhang, Q. Chen, D. S. Kirschen, P. Li, and Q. Xia, "Data-driven probabilistic net load forecasting with high penetration of behind-the-meter pv," *IEEE Transactions on Power Systems*, vol. 33, no. 3, pp. 3255–3264, 2017.

[27] A. Kaur, L. Nonnenmacher, and C. F. Coimbra, "Net load forecasting for high renewable energy penetration grids," *Energy*, vol. 114, pp. 1073–1084, 2016.

[28] A. Stratman, T. Hong, M. Yi, and D. Zhao, "Net load forecasting with disaggregated behind-the-meter pv generation," *IEEE Transactions on Industry Applications*, 2023.

[29] P. Kobylinski, M. Wierzbowski, and K. Piotrowski, "High-resolution net load forecasting for micro-neighbourhoods with high penetration of renewable energy sources," *International Journal of Electrical Power & Energy Systems*, vol. 117, p. 105635, 2020.

[30] G. Tziolis, A. Koumis, S. Theocharides, A. Livera, J. Lopez-Lorente, G. Makrides, and G. E. Georghiou, "Advanced short-term net load forecasting for renewable-based microgrids," in *2022 IEEE International Smart Cities Conference (ISC2)*. IEEE, 2022, pp. 1–6.

[31] F. Kabir, N. Yu, W. Yao, R. Yang, and Y. Zhang, "Joint estimation of behind-the-meter solar generation in a community," *IEEE Transactions on Sustainable Energy*, vol. 12, no. 1, pp. 682–694, 2020.

[32] D. Chen and D. Irwin, "Sundance: Black-box behind-the-meter solar disaggregation," in *Proceedings of the eighth international conference on future energy systems*, 2017, pp. 45–55.

[33] F. Kabir, N. Yu, W. Yao, R. Yang, and Y. Zhang, "Estimation of behind-the-meter solar generation by integrating physical with statistical models," in *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE, 2019, pp. 1–6.

[34] K. Li, F. Wang, Z. Mi, M. Fotuhi-Firuzabad, N. Duić, and T. Wang, "Capacity and output power estimation approach of individual behind-the-meter distributed photovoltaic system for demand response baseline estimation," *Applied energy*, vol. 253, p. 113595, 2019.

[35] C. M. Cheung, S. R. Kuppannagari, A. Srivastava, R. Kannan, and V. K. Prasanna, "Behind-the-meter solar generation disaggregation at varying aggregation levels using consumer mixture models," *IEEE Transactions on Sustainable Computing*, 2022.

[36] M. Khodayar, J. Wang, and Z. Wang, "Energy disaggregation via deep temporal dictionary learning," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 5, pp. 1696–1709, 2019.

[37] F. Bu, K. Dehghanpour, Y. Yuan, Z. Wang, and Y. Zhang, "A data-driven game-theoretic approach for behind-the-meter pv generation disaggregation," *IEEE Transactions on Power Systems*, vol. 35, no. 4, pp. 3133–3144, 2020.

[38] W. Li, M. Yi, M. Wang, Y. Wang, D. Shi, and Z. Wang, "Real-time energy disaggregation at substations with behind-the-meter solar generation," *IEEE Transactions on Power Systems*, vol. 36, no. 3, pp. 2023–2034, 2020.

[39] C. Tian and W. K. Chan, "Spatial-temporal attention wavenet: A deep learning framework for traffic prediction considering spatial-temporal dependencies," *IET Intelligent Transport Systems*, vol. 15, no. 4, pp. 549–561, 2021.

[40] F. Bu, K. Dehghanpour, Y. Yuan, Z. Wang, and Y. Zhang, "A data-driven game-theoretic approach for behind-the-meter pv generation disaggregation," *IEEE Transactions on Power Systems*, vol. 35, no. 4, pp. 3133–3144, 2020.

[41] E. Rossi, H. Kenlay, M. I. Gorinova, B. P. Chamberlain, X. Dong, and M. M. Bronstein, "On the unreasonable effectiveness of feature propagation in learning on graphs with missing node features," in *Learning on Graphs Conference*. PMLR, 2022, pp. 11–1.

[42] E. Arazo, D. Ortego, P. Albert, N. O'Connor, and K. McGuinness, "Unsupervised label noise modeling and loss correction," in *International conference on machine learning*. PMLR, 2019, pp. 312–321.

[43] B. Zhang, Y. Li, Y. Tu, J. Peng, Y. Wang, C. Wu, Y. Xiao, and C. Zhao, "Learning from noisy labels with coarse-to-fine sample credibility modeling," in *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*. Springer, 2023, pp. 21–38.

[44] T. Moon, "The expectation-maximization algorithm," *IEEE Signal Processing Magazine*, vol. 13, no. 6, pp. 47–60, 1996.

[45] K. Nagasawa, C. R. Upshaw, J. D. Rhodes, C. L. Holcomb, D. A. Walling, and M. E. Webber, "Data management for a large-scale smart grid demonstration project in austin, texas," in *Energy Sustainability*, vol. 44816. American Society of Mechanical Engineers, 2012, pp. 1027–1031.

[46] Nationwide analysis of u.s. commercial building solar photovoltaic (pv) breakeven conditions. [Online]. Available: https://www.nrel.gov/docs/fy16osti/64793.pdf

[47] K. Li, J. Yan, L. Hu, F. Wang, and N. Zhang, "Two-stage decoupled estimation approach of aggregated baseline load under high penetration of behind-the-meter pv system," *IEEE Transactions on Smart Grid*, vol. 12, no. 6, pp. 4876–4885, 2021.

[48] Y. Hu, X. Cheng, S. Wang, J. Chen, T. Zhao, and E. Dai, "Times series forecasting for urban building energy consumption based on graph convolutional network," *Applied Energy*, vol. 307, p. 118231, 2022.

[49] L. Chen, D. Chen, Z. Shang, B. Wu, C. Zheng, B. Wen, and W. Zhang, "Multi-scale adaptive graph neural network for multivariate time series forecasting," *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[50] Y. Liu, H. Wu, J. Wang, and M. Long, "Non-stationary transformers: Exploring the stationarity in time series forecasting," *Advances in Neural Information Processing Systems*, vol. 35, pp. 9881–9893, 2022.

SUPPLEMENTARY MATERIAL

A. Improved BMM Algorithm

Algorithm 1 Improved BMM

1: Initialization: $p(k), \alpha_k, \beta_k, \quad k = 1, 2$; maximum iterations B.
2: Estimate α_1, β_1 using the forecasting loss variances in C^F and C^N buildings.
3: Fix α_1, β_1, and use EM algorithm to estimate $p(k), \alpha_2, \beta_2$ using forecasting loss variances in C^P buildings,
4: **repeat**
5: E-step: calculate the responsibility weight $w_k(l_{\text{var}})$ by

$$w_k(l_{\text{var}}) = \frac{p(k)(p(l_{\text{var}}|k=1))}{\sum_{k=1}^{2} p(k)p(l_{\text{var}}|k)}. \tag{22}$$

6: M-step: (i) update the prior probability $p(k)$ by

$$p(k) = |\Omega_d|^{-1}|C^P|^{-1} \sum_{t\in\Omega_d, i\in C^P} w_k(l_{\text{var\_}ti}); \tag{23}$$

(ii) update α_2, β_2 by

$$\alpha_2 = \frac{m_2^2(1-m_2)}{s_2}, \beta_2 = \frac{\alpha_2(1-m_2)}{m_2} \tag{24}$$

where m_2 and s_2 are the weighted mean values and variances of forecasting loss variances in C^P buildings, respectively. m_2 and s_2 can be calculated by

$$\begin{aligned} m_2 &= \frac{\sum w_2(l_{\text{var\_}ti}) l_{\text{var\_}ti}}{\sum w_2(l_{\text{var\_}ti})} \\ s_2 &= \frac{\sum w_2(l_{\text{var\_}ti})(l_{\text{var\_}ti} - m_2)^2}{\sum w_2(l_{\text{var\_}ti})} \end{aligned} \tag{25}$$

7: **until** The number of iterations is up to B.
8: Calculate γ by equation (15) and (16).

Output: γ

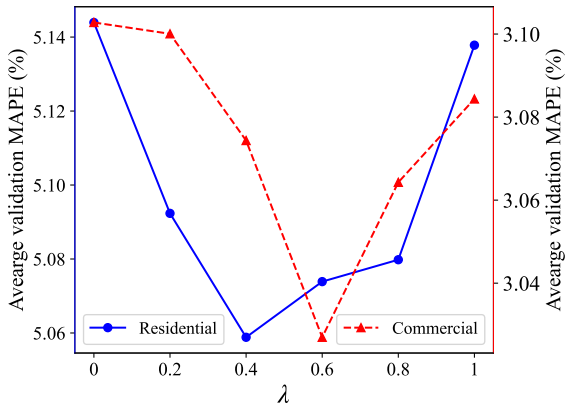

Fig. 11. λ selection in ULoFo

B. Hyperparameter Selection

The hyperparameters of the forecasting component in ULoFo follow the parameter settings in the paper that originally proposed it [39]. Specifically, the input time window size K is set to 12. The total number of neural network architecture layers is 10, and the number of hidden nodes in every layer is set as $(32, 256, 256, 256, 256, 256, 256, 256, 256, 1)$. The number of epochs is 100. The dimension of the trainable query matrix W_q and key matrix W_k are $\mathbb{R}^{48\times16}$, respectively. The size of the learnable node embedding e_i is 16. Besides the hyperparameters of the forecasting component in ULoFo, only one regularization coefficient λ in the disaggregation component GSD will impact the performance of ULoFo and need to be selected carefully. This paper leverages the forecasting validation error to select the coefficient λ. As shown in Fig.11, the optimal λ in the residential experiment is 0.4, and the optimal λ in the commercial experiment is 0.6.

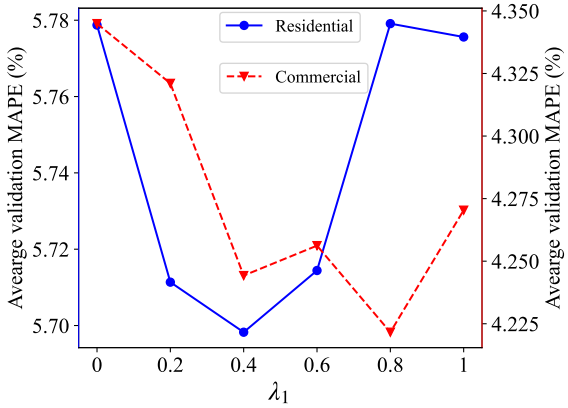

Fig. 12. λ_1 selection in CMM-STGCN

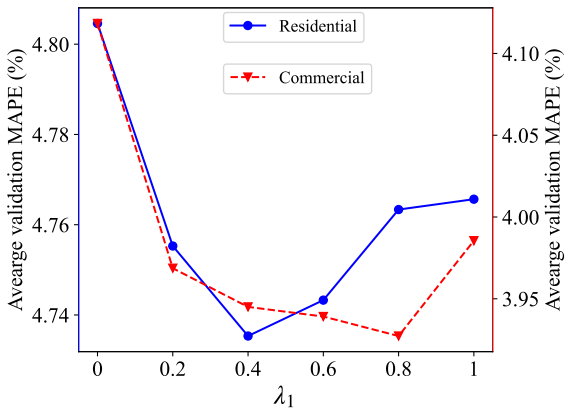

Fig. 13. λ_1 selection in CMM-GW

To provide a fair comparison between the proposed ULoFo and the baseline CMM-STGCN, the hyperparameters of the forecasting component in CMM-STGCN also follow the parameter settings in the papers that originally proposed it [48]. Specifically, the input window size is also set to 12, and the number of training epochs is 100. The total number of neural network layers in CMM-STGCN is 9, and the number of hidden nodes in each layer is $(64, 16, 64, 64, 16, 64, 128, 128, 1)$,

TABLE VI
COMPARISONS AMONG ULOFO AND BASELINES ON RESIDENTIAL BUILDINGS

Methods	Disaggregation						Prediction			Computation time	
	Load error			Solar error			Test error			Offline	Real time
	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE		
CMM-STGCN	0.145	6.51%	0.269	0.145	13.34%	0.269	0.203	8.7%	0.344	437.6	6.9
CMM-GW	0.145	6.51%	0.269	0.145	13.34%	0.269	0.166	7.11%	0.3	1366.1	5.8
ULoFo (ours)	**0.109**	**4.86%**	**0.282**	**0.109**	**10.12%**	**0.358**	**0.152**	**6.51%**	**0.292**	3921.3	9.8

respectively. In the disaggregation component CMM of CMM-STGCN, there is a sensitive hyperparameter, the weighted term λ_1 between night time only and full-time periods, needs to be selected by prediction validation error. As shown in Fig. 12, the optimal λ_1 in the residential experiment is 0.4, and the optimal λ_1 in the commercial experiment is 0.8. Similarly, the hyperparameters of the forecasting component in CMM-GW follow the parameter settings in [14]. Specifically, the input time window length is set to 12, and the number of training epochs is also set to 100. Since CMM-GW has the same disaggregation component as CMM-STGCN, the same parameter λ_1 needs to be determined by the prediction validation error of CMM-GW. As shown in Fig. 13, the optimal λ_1 of CMM-GW in the residential and commercial experiments are 0.4 and 0.8, respectively.

In the prediction only comparison, the parameter settings of the five methods, GraphWavenet, STGCN, MAGNN, NTS, and STAWnet, follow the parameter settings in [14], [48], [49], [50], and [39], respectively. In the disaggregation only comparison, the baseline TLA includes one regularization term λ_2 in the optimization objective, which is demonstrated to be robust in the range $(100, 500)$ in [4]. Thus, we set λ_2 to 100 in this paper. The insensitive regularization coefficient in DL-SD is set to 0.1 in this paper. In the baseline CMM and the proposed GSD, as discussed before, the hyperparameter λ_1 in CMM is set to 0.4 and 0.8 in the residential and commercial experiments, respectively. The hyperparameter λ in GSD is set to 0.4 and 0.6 in the residential and commercial experiments, respectively.

C. Result Analysis on residential buildings

1) Dataset description: The native load values of 30 residential buildings are randomly selected from a Typical Meteorological Year (TMY3) dataset in Texas [19]. We randomly select 10 buildings, 6 buildings, and 14 buildings from the 30 residential buildings to construct the residential building set C^N, C^F, and C^P, respectively. The PV generations of 20 sites in Texas are randomly selected from a public dataset in [45] for the experiment on residential buildings. The residential PV generations are added to the native load values in residential buildings of C^F and C^P to obtain the net load values. The number of residential buildings in C^N, C^F, and C^P are 10, 6, and 14, respectively. Both the residential load profile and PV generation profile have 8760 time steps with one-hour resolution in one year. The one-year load data are divided into training, validation, and test datasets by the ratio of 0.7 : 0.2 : 0.1.

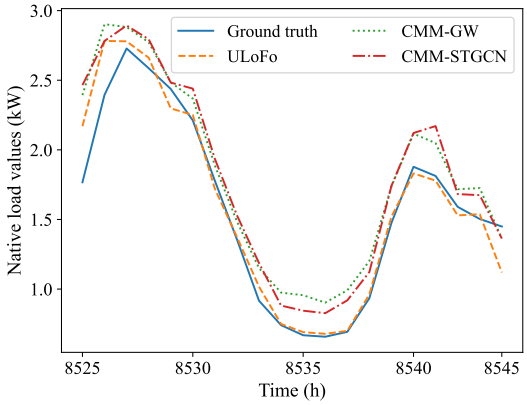

Fig. 14. Prediction results of one residential building

TABLE VII
LOAD PREDICTION PERFORMANCE AT RESIDENTIAL BUILDINGS WHERE THE NATIVE LOAD IS MEASURED AND NOT MEASURED

Methods	Load measured			Load not measured		
	MAE	MAPE	RMSE	MAE	MAPE	RMSE
CMM-STGCN	0.171	7.24%	0.297	0.24	10.36%	0.39
CMM-GW	0.12	5.04%	0.234	0.219	9.47%	0.362
ULoFo (ours)	**0.109**	**4.58%**	**0.216**	**0.201**	**8.72%**	**0.359**

2) Performance Evaluation and Comparison: As shown in Table VI, the proposed ULoFo performs the best in both residential load disaggregation and prediction tasks. ULoFo has the highest offline computational cost. The real-time computational cost of ULoFo is slightly higher than the cost of CMM-STGCN and CMM-GW. One can also see the predicted residential load values by ULoFo are the closest to the ground truth values in Fig. 14. As shown in Table VII, ULoFo can also achieve the best performance in both cases, where the native load is directly observed and is not observed.

3) Performance of Disaggregation or Forecasting Only: For disaggregation only, we compare the GSD component of ULoFo with three other existing disaggregation methods. As shown in Table VIII, the disaggregation performance of CMM is slightly better than that of GSD. The proposed GSD performs better than TLA and DL-SD. The reason for the poor performance of TLA may be a significant error, which achieves around 12.9%, in the estimation of PV peak generation. Overall, the performance of GSD can be comparable with the existing disaggregation baselines. Furthermore, GSD has the lowest computational cost in residential load disaggregation.

We then compare the prediction component STAWnet of ULoFo with four other four prediction approaches: Graph-Wavenet, STGCN, MAGNN, and NTS. As shown in Table IX, the performance of STAWnet is the best among the five prediction approaches regarding real-time residential load forecasting. Similar to the analysis in the commercial building experiment, NTS and MAGNN perform poorly because the two large models may not be fully trained. Compared with the relatively small STGCN model, GraphWavenet has more spatial-temporal modules, and thus shows better performance than STGCN. Compared to GraphWavenet, the only difference between STAWnet and GraphWavenet is that it can learn the dynamic spatial correlation among the input data. Therefore, STAWnet can achieve better performance than GraphWavenet.

TABLE VIII
DISAGGREGATION ONLY PERFORMANCE WITH DIRECTLY MEASURED RESIDENTIAL LOAD

Methods	Load error			Solar error			Time (s)
	MAE	MAPE	RMSE	MAE	MAPE	RMSE	
TLA	0.182	7.94%	0.436	0.182	16.06%	0.436	164.44
CMM	**0.145**	**6.51%**	**0.269**	**0.145**	**13.34%**	**0.269**	9.1
DL-SD	0.215	9.46%	0.488	0.215	19.01%	0.488	493.82
GSD (ours)	0.153	6.78%	0.382	0.153	15.77%	0.382	**0.34**

TABLE IX
PREDICTION ONLY PERFORMANCE WITH DIRECTLY MEASURED RESIDENTIAL LOAD

Methods	Load prediction error			Parameter
	MAE	MAPE	RMSE	
GraphWavenet	0.116	4.94%	0.233	267449
STGCN	0.166	7.04%	0.291	153985
MAGNN	0.24	10.46%	0.43	1302885
NTS	0.279	11.98%	0.455	10810718
STAWnet (ours)	**0.107**	**4.55%**	**0.215**	271681

TABLE X
RESIDENTIAL DISAGGREGATION AND PREDICTION PERFORMANCE THROUGH REFURBISHMENT

Times	γ	Disaggregation		Prediction	Inference Time (s)
		Load MAPE	Solar MAPE	Load MAPE	
0	-	5.11%	11.74%	6.62%	7.1
1	0.65	4.91%	11.34%	-	-
2	0.85	4.89%	11.29%	-	-
3	0.89	**4.86%**	**11.23%**	**6.51%**	9.8

4) Performance of Graph sparsification: Similarly, the pruning rate r is varied from 0 to 0.9 with a step size of 0.15. Fig. 15 shows how the validation MAPE and test MAPE of load prediction change. One can see that both curves have the same trend of decreasing first for a large range of r until $r = 0.45$ and then increasing for a very large pruning rate. This justifies our strategy of using the validation MAPE as the criterion to select the best r. Fig. 16 also shows the disaggregation MAPE, both of which have the same trend as forecasting MAPE.

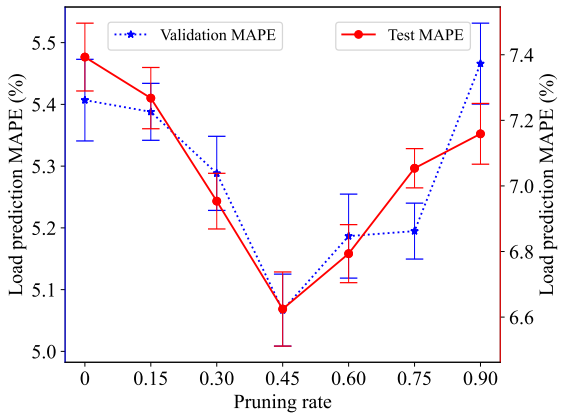

Fig. 15. Residential prediction performance under different pruning rate

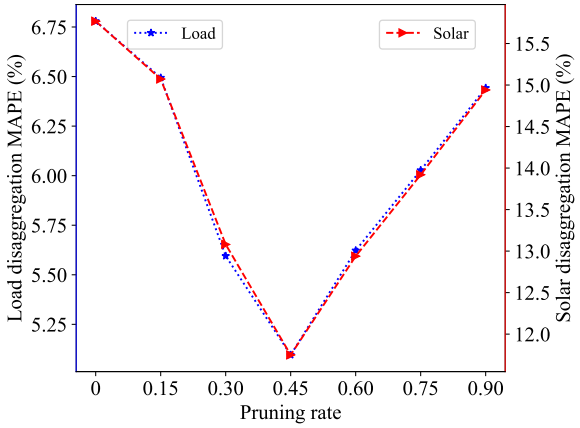

Fig. 16. Residential disaggregation performance under different pruning rate

5) Performance of input refurbishment: From Table X, one can see both the disaggregation accuracy and prediction accuracy improve after three times refurbishments, which then demonstrates the effectiveness of the input refurbishment strategy. As mentioned before, γ_i reflects the probability of the currently estimated input being accurate. One can see that γ_i increases during the refurbishment process, and so does the accuracy of the estimated result. Furthermore, from $i = 2$ to 3, γ_i does not change significantly, and the estimated result has a minor improvement. Following the rule discussed in the commercial building experiment for determining the total number of refurbished times I, I is set to be 3 in this experiment.

6) Robustness of the interaction strategy: In this paper, the two strategies, graph sparsification, and input refurbishment, use the forecasting result to help improve disaggregation performance, which in turn boosts prediction performance. That means the disaggregation module is affected by the forecasting module by the two strategies. Here, we verify the robustness of the two interaction strategies relative to forecasting performance. We adjusted the parameters of the prediction model STAWnet in the offline training stage to obtain multiple prediction models STAWnet with different levels of accuracy. The performance of the two interaction

strategies is discussed under varying forecasting accuracy levels.

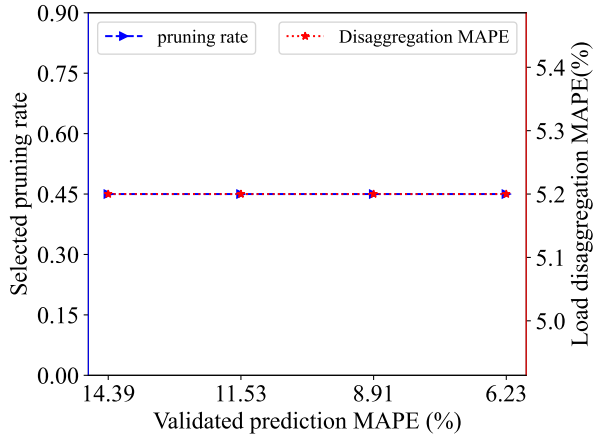

Fig. 17. Selected pruning rate and disaggregation performance

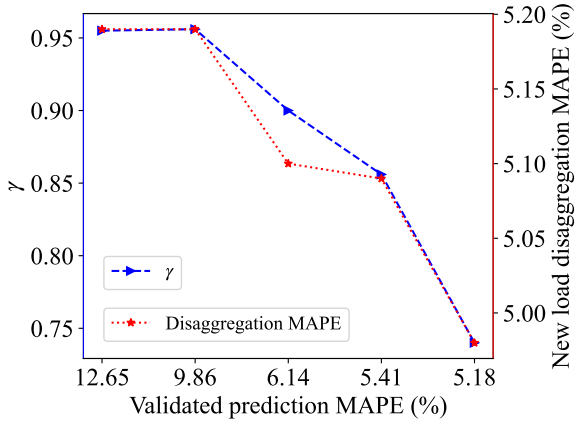

Fig. 18. Input refurbishment under different forecasting performance

In the strategy of graph sparsification, Fig. 17 shows the selected optimal pruning rate by the prediction model when the model and the corresponding prediction accuracy vary. One can see that even when the prediction accuracy decreases, the selected pruning rate remains 0.45, which is verified as the optimal pruning rate in Fig. 16. Thus, the graph sparsification is robust for forecasting accuracy.

In the input refurbishment, the weight γ, computed by (16), can help prevent error propagation from poor forecasting performance. As shown in Fig. 18, γ remains a large value when forecasting performance is inaccurate. That means a large weight is assigned to the original estimation $\tilde{X}^P$ on the RHS of (11), and thus, the forecasting errors from $\hat{y}$ will not be propagated to the new estimation. As forecasting performance improves, γ decreases, and a larger weight will be assigned to $\hat{y}$ on the RHS of (11), which thus helps improve the performance of new estimated $\tilde{X}^P$. Therefore, the input refurbishment is robust to the forecasting performance with the help of varying γ.