Bayesian Energy Disaggregation at Substations with Uncertainty Modeling

Ming Yi, *Student Member, IEEE,* Meng Wang, *Member, IEEE*

***Abstract*—This paper considers energy disaggregation at substations (EDS) where the objective is to estimate the consumption of each load from aggregate measurements, in which whether or not some loads are consuming power is unknown to the operator. The existing EDS method cannot provide any reliability measure of the disaggregation results, while the disaggregation accuracy can vary significantly for different data due to the volatility of loads such as the solar generation. This paper proposes a Bayesian-dictionary-learning-based approach to disaggregate the loads and provides an uncertainty measure of the returned estimation. Our approach learns the probability distributions of the load patterns and the decomposition coefficients from recorded data with partial labels at the offline stage. In real-time disaggregation of the obtained aggregate data, our approach computes the mean and covariance of the probability distribution of each load consumption, estimates the load using the mean, and computes the uncertainty index based on the covariance. Numerical experiments indicate that our method achieves improved disaggregation accuracy over the existing EDS method, and the uncertainty index measures the reliability of the returned estimation accurately.**

***Index Terms*—Energy disaggregation, behind-the-meter solar generation, Bayesian dictionary learning, uncertainty modeling**

I. Introduction

At a distribution substation, measurements are taken constantly about the net power consumption of all the loads[1], such as residential loads, industrial loads, wind and solar generations, while the power consumption of individual load types is not directly measured. Energy disaggregation at the substation level (EDS) wants to extract the energy consumption of each type of load from the aggregated net load measurements. The accurate consumption of each load type is useful for distribution system planning and operations, such as hosting capacity evaluation [1], [2], providing restoration solutions for distribution networks [3], [4], net load forecasting [5], [6], demand response and load dispatching [7], [8] and dynamic Volt/Var control [9], [10]. EDS becomes increasingly challenging due to the volatility of renewable generations such as the behind-the-meter (BTM) solar generations.

EDS is different from energy disaggregation at the household level (EDH) [11]–[14], which is studied under the terminology of non-intrusive load monitoring (NILM) [11]–[17]. Most household-level electric appliances demonstrate repeatable characteristics and are often single-sate or multi-state devices. EDH methods first estimate the patterns of individual appliances from measurements that contain one appliance only and then identify appliances from the testing data using these patterns. At the substation level, in contrast, the power consumption is highly aggregated from various types of loads. It is more challenging to obtain measurements that only contain one type of load at the substation than the household level and, thus, more difficult to learn the distinctive patterns of different loads. In addition, although the operator knows the types of loads connected to a substation, it may not know whether all the loads are consuming power or not in a given time window. For instance, the operator does not have direct information about whether the BTM solar generation is functioning or not at a particular time. In fact, identifying the actual amount of generation by the BTM solar is an important and challenging task. In theory, one can disaggregate the solar generation at each household from the smart meter and add them to obtain the total solar generation [18]. If smart meters are not available at every home or there is a large number of houses, solving EDS directly using the aggregate measurements is a practical and computationally efficient approach.

Ref. [19] is the first work to formulate the EDS as a deterministic dictionary learning problem from so-called “partially labeled aggregate data.” Note that due to the additive features in power consumption, the problem formulated by [19] differs from the conventional dictionary learning using incomplete label information in image classification [20], [21]. Because some loads such as BTM solar are volatile, and load patterns are masked in the aggregated measurements and difficult to learn accurately, it is natural that the estimated power consumption contains errors. The method in [19], however, only provides an estimate of the power consumption of each type of load and does not provide any information about the reliability of the estimation. For the operator to make an informative decision, it is imperative to develop EDS methods that can provide an uncertainty measure of the returned estimation. Moreover, the deterministic approach in [19] does not model measurement noise explicitly, and the disaggregation performance degrades significantly when the noise level is high.

This paper develops a probabilistic EDS method to estimate the consumption of different load types from aggregate measurements and computes an uncertainty measure of the disaggregation results. The proposed method includes both the offline learning that estimates the load patterns, referred to as dictionaries, from recorded noisy training data and the online

This work was supported in part by the NSF grant # 1932196, AFOSR FA9550-20-1-0122, and ARO W911NF-21-1-0255. (Corresponding author: Meng Wang.)

M. Yi, and M. Wang are with the Dept. of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY. Email:{ yim3, wangm7}@rpi.edu.

[1] Renewable generation is also referred to as a negative “load” in this paper.

disaggregation that separates the consumption of each type of load from the aggregate power consumption. The offline learning problem is formulated as a Bayesian dictionary learning problem where given prior distributions of model parameters and the recorded data, we compute the posterior distributions of the dictionaries. In online disaggregation, we use the learned distributions of the dictionaries to estimate the probability distributions of the power consumption of each load. We also compute an uncertainty index based on the singular values of the covariance matrix of the estimated distribution. To the best of our knowledge, this is the first work on EDS that provides an uncertainty measure, which can be used to evaluate the reliability of the disaggregation result. Moreover, our method has a much higher disaggregation accuracy than the deterministic approach in [19], especially when the measurements are noisy. Furthermore, the performance of the deterministic dictionary learning methods depends critically on the prior selection of model parameters such as the dictionary size, and the performance degrades significantly if the parameters are not selected properly. In contrast, our method based on Bayesian dictionary learning can learn the dictionary size from data and is also more robust to other model parameters.

Bayesian dictionary learning [22]–[24] has been exploited in applications like image denoising and object classification. To the best of our knowledge, this paper is the first work that studies the EDS problem from the perspective of Bayesian dictionary learning. Moreover, conventional Bayesian dictionary learning methods require that every training data point belongs to exactly one class and cannot handle the case when a training data point is the sum of multiple types of loads. This paper provides the first formulation and solution of Bayesian dictionary learning from partially labeled aggregate data. Our approach learns the dictionaries of different types of loads from aggregate data, even when the load types in each training data sample are not fully known. This is a general methodology and can be applied to other domains beyond energy disaggregation, as long as the data are additive of different classes.

A. Related Works

Both model-based and data-driven methods have been developed to estimate solar loads from the net point. The model-based methods [18], [25]–[27] utilize parametric models to estimate BTM solar generation. The estimation accuracy largely depends on the precise meteorological and geographic information, which are not always available and may be expensive to obtain. Data-driven methods [28]–[33] usually require high-resolution historical load profiles. These two categories of methods disaggregate solar load only but not other types of loads. Refs. [34], [35], [36], [37] disaggregate different types of loads from the feeder level at the substation, but they develop parametric models for each load and require weather information.

The non-intrusive load monitoring [11]–[17] studies the energy disaggregation problem at the household level. Most of existing methods require recorded measurements of individual appliances to learn the typical features. Various approaches have been proposed such as training deep neural networks [13], [38], [39], modeling NILM as factorial hidden Markov model [12], [40], [41], learning representative patterns by dictionary learning [11], [42], [43], constraining the states of electrical appliances by mixed-integer linear programming [44], and solving matrix decomposition problem [45], [46]. Among them, hidden-Markov-model-based algorithms model the discrete operating states of household electrical appliances probabilistically and often require high temporal data resolution. At the substation level, the sampling rate is not high, and the complexity of modeling individual appliances at each household quickly explodes as the number of users increases. Dictionary-learning-based approaches are model-free and computationally efficient. The disaggregation accuracy can be improved by promoting discriminative dictionaries [11], [42], [43] and adding regularization constraints [45]. None of these works can characterize the uncertainty of solutions.

Only a few works model the uncertainty of the prediction. Ref. [6] uses Bayesian neural networks to forecast residential loads and characterize the uncertainty of the forecasting results. Ref. [32] proposes to model uncertainty of solar generation by the fuzzy interval. These methods forecast one load based on the historical data and do not consider disaggregating multiple loads from aggregate measurements.

II. Problem Formulation

There are C ($C > 1$) types of loads in total connecting to a substation. A load is considered as a positive load if it consumes the power or a negative load if it generates power. Examples include but not limited to residential load, industrial load, solar generation and wind generation. Note that in a given time interval, not every load is consuming/generating power. Let $\boldsymbol{x} \in \mathbb{R}^P$ denote the total power consumption at the substation during a time interval with length P. $\boldsymbol{y} = [y^1, y^2, ..., y^C] \subseteq \{0,1\}^C$ is a multi-label binary vector with size C that indicates the load types that exist in $\boldsymbol{x} \in \mathbb{R}^P$, i.e., $y^c = 1$ if load c is nonzero in $\boldsymbol{x}$, and $y^c = 0$ of load c is zero in $\boldsymbol{x}$. For example, when $C = 3$, $\boldsymbol{y} = [0, 1, 1]^T$ indicates that loads 2 and 3 exist in $\boldsymbol{x}$ but not load 1.

Only partial entries in $\boldsymbol{y}$ are known to the operator, while whether or not some other loads exist in a specific time series $\boldsymbol{x}$ is unknown. This setup follows the “partially labeled aggregate data” in [19]. As described in [19], the partial labels can be obtained either by engineering experience (e.g., residential load exists during 7-9 pm) or applying a detector for some types of load [47], [48]. Because the measurements are aggregated at the substation, a detector may fail to detect the existence of some loads [49], resulting in partial labels. We also remark that the scenario when all the labels are known is a special case of our setup, and the proposed method in this paper for partial labels naturally handles full labels as well.

Let $\boldsymbol{X} \in \mathbb{R}^{P \times N}$ represent N recorded measurements, each with window length P. The ith column of $\boldsymbol{X}$, denoted by $\boldsymbol{x}_i$, represents the data at the ith interval. Let $\boldsymbol{y}_i$ denote the corresponding multi-label of $\boldsymbol{x}_i$. The $C \times N$ matrix $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_N]$ collects all the labels. Let Ω denote the set of indices where entries of $\boldsymbol{Y}$ are known. Let $\bar{\Omega}$ be the complement set of Ω. Then the index (c, n) belonging to Ω means that we know whether load c is in $\boldsymbol{x}_n$ or not. Otherwise, (c, n) belongs to $\bar{\Omega}$. The partially known labels are characterized in $\boldsymbol{Y}_\Omega$. In the above example where $N = 1$ and $C = 3$, if one only knows load 2 exists in $\boldsymbol{x}$, and no information is provided about load 1 and 3, then $\boldsymbol{Y}_\Omega$ can be written as $\boldsymbol{Y}_\Omega = [?, 1, ?]^T$ where ? denotes the entries not in Ω.

Then given a time series of aggregate measurement $\hat{\boldsymbol{x}} \in \mathbb{R}^P$, the questions this paper addresses are (1) what is the corresponding power consumption of each load c in $\hat{\boldsymbol{x}}$, denoted by $\hat{\boldsymbol{x}}^c$? (2) What is the uncertainty of this estimation?

III. Bayesian Energy Disaggregation using Partially Labeled Aggregate Data

Following the dictionary learning framework, the dictionary $\boldsymbol{D}^c \in \mathbb{R}^{P \times K_c}$ contains K_c representative patterns of load c. Then the aggregate data $\boldsymbol{x}_i$ can be written as $\boldsymbol{x}_i = \sum_{c=1}^{C} \boldsymbol{D}^c \boldsymbol{\omega}_i^c + \boldsymbol{\epsilon}_i$, where $\boldsymbol{\omega}_i^c \in \mathbb{R}^{K_c}$ represents the coefficients for load c, and $\boldsymbol{\epsilon}_i$ represents the noise.

Fig. 1 is an overview of our proposed approach. In the offline training stage (Section III-A), the method infers K_c and learn the distributions of the dictionaries $\boldsymbol{D}^c$ and the coefficients from the recorded data $\boldsymbol{X}$ and the corresponding partial labels $\boldsymbol{Y}_\Omega$. In online disaggregation (Section III-B), based on the learned probabilistic model, the method computes the distribution of $\hat{\boldsymbol{x}}^c$, which is the consumption of load c in $\hat{\boldsymbol{x}}^c$. The mean will be used as an estimate of $\hat{\boldsymbol{x}}^c$, and the uncertainty measure is calculated based on the covariance.

A. Bayesian dictionary learning from partially labeled aggregate data

A probabilistic model is employed to describe the data. The hierarchical model is shown in equations (1) to (11), and visualized in Fig. 2. Conventional Bayesian dictionary learning uses data belonging to one same class to learn one dictionary. This model here extends from the model in conventional dictionary learning from two aspects. First, aggregate data from C dictionaries rather than one dictionary are considered here, as shown in (1). Second and more importantly, the load types that actually exist in each training sample $\boldsymbol{x}_i$ are not fully known. Different from conventional Bayesian dictionary learning, one needs to additionally estimate the full labels (load types that actually exist) of each training data point. Thus, a new binary variable y_i^c is introduced here as an indicator of the existence of load c in training sample i.

For all $i = 1, 2, 3, ..., N$, $c = 1, 2, 3, ..., C$, and $k = 1, 2, 3, ..., K_c$,

$$\boldsymbol{x}_i = \sum_{c=1}^{C} \boldsymbol{D}^c \boldsymbol{\omega}_i^c + \boldsymbol{\epsilon}_i \tag{1}$$

$$\boldsymbol{\omega}_i^c = (\boldsymbol{z}_i^c \odot \boldsymbol{s}_i^c) y_i^c \tag{2}$$

$$\boldsymbol{d}_k^c \sim \mathcal{N}(\boldsymbol{0}, \frac{1}{\lambda_d} \boldsymbol{I}_P) \tag{3}$$

$$\boldsymbol{z}_i^c \sim \prod_{k=1}^{K_c} \text{Bernoulli}(\pi_k^c) \tag{4}$$

$$\pi_k^c \sim \text{Beta}(\frac{a_0}{K_c}, \frac{b_0(K_c - 1)}{K_c}) \tag{5}$$

$$\boldsymbol{s}_i^c \sim \mathcal{N}(\boldsymbol{0}, \frac{1}{\gamma_s^c} \boldsymbol{I}_{K_c}) \tag{6}$$

$$\boldsymbol{\epsilon}_i \sim \mathcal{N}(\boldsymbol{0}, \frac{1}{\gamma_\epsilon} \boldsymbol{I}_P) \tag{7}$$

$$\gamma_s^c \sim \Gamma(c_0, d_0) \tag{8}$$

$$\gamma_\epsilon \sim \Gamma(e_0, f_0) \tag{9}$$

$$\boldsymbol{y}_i \sim \prod_{c=1}^{C} \text{Bernoulli}(\phi^c) \tag{10}$$

$$\phi^c \sim \text{Beta}(g_0, h_0) \tag{11}$$

Each column of the dictionary $\boldsymbol{D}^c$, denoted by $\boldsymbol{d}_k^c$ for the kth column, is sampled from a multi-variant Gaussian $\mathcal{N}(\boldsymbol{0}, \frac{1}{\lambda_d} \boldsymbol{I}_P)$, where λ_d is a fixed constant, and $\boldsymbol{I}_P$ is a $P \times P$ identity matrix. The vector $\boldsymbol{\epsilon}_i$ models the measurement noise and is from Gaussian $\mathcal{N}(\boldsymbol{0}, \frac{1}{\gamma_\epsilon} \boldsymbol{I}_P)$. The coefficient vector $\boldsymbol{\omega}_i^c$ includes the corresponding coefficients in training sample $\boldsymbol{x}_i$ of all dictionary atoms of $\boldsymbol{D}^c$. As shown in (2), $\boldsymbol{\omega}_i^c$ can be viewed as the point-wise product of two vectors $\boldsymbol{z}_i^c$ and $\boldsymbol{s}_i^c$ in $\mathbb{R}^{K_c}$ and then multiplied by the value y_i^c.

y_i^c is binary scalar indicating the existence of load c in $\boldsymbol{x}_i$. If y_i^c is zero, $\boldsymbol{\omega}_i^c$ is a zero vector. y_i^c is sampled from the Bernoulli distribution with ϕ^c. ϕ^c governs the probability that load c exists in the aggregate data and is drawn from a Beta distribution with pre-determined constants g_0 and h_0.

$\boldsymbol{z}_i^c$ is a binary vector and its kth entry, denoted by $\boldsymbol{z}_{ik}^c$ indicates whether the pattern represented by the kth dictionary atom of $\boldsymbol{D}^c$ exists in $\boldsymbol{x}_i$ or not. If $y_i^c = 1$ and $\boldsymbol{z}_{ik}^c = 0$, that means load c exists in $\boldsymbol{x}_i$ but the specific pattern $\boldsymbol{d}_k^c$ does not exist in $\boldsymbol{x}_i$. The kth entry of $\boldsymbol{z}_i^c$ is sampled from the Bernoulli distribution with π_k^c. π_k^c governs the probability of the existence of $\boldsymbol{d}_k^c$ in the aggregate data. π_k^c is drawn from a Beta distribution with pre-determined constants a_0 and b_0. As shown in [22], $\boldsymbol{z}_i^c$ generated from this process is sparse, i.e., contains many zero entries. After computing the posterior distributions of π_k^c and $\boldsymbol{z}_i^c$ using the recorded data, one can prune the dictionaries based on $\boldsymbol{z}_i^c$. Therefore, K_c are set as large values in the prior distribution, and the actual dictionary sizes are learned from the data.

The coefficient vector $\boldsymbol{s}_i^c$ is sampled from $\mathcal{N}(\boldsymbol{0}, \frac{1}{\gamma_s^c} \boldsymbol{I}_{Kc})$. Two gamma priors are placed on γ_s^c and γ_ϵ with pre-determined constants c_0, d_0, e_0 and f_0. Note that the priors in our model are all conjugate priors, which can simplify the computation of posterior in the following discussion.

Let $\boldsymbol{\Theta} = \{\boldsymbol{d}_k^c, \boldsymbol{z}_i^c, \boldsymbol{s}_i^c, \boldsymbol{\pi}_k^c, \boldsymbol{\phi}^c, \gamma_s^c, \gamma_\epsilon, i = 1, 2, 3, ..., N, c = 1, 2, 3, ..., C, k = 1, 2, 3, ..., K_c\}$ denote all the latent variables. Given $\boldsymbol{X}$ and partial labels $\boldsymbol{Y}_\Omega$, the goal is to compute the posterior $P(\boldsymbol{\Theta}, \boldsymbol{Y}_{\bar{\Omega}} | \boldsymbol{X}, \boldsymbol{Y}_\Omega)$. From the Bayes rule,

$$P(\boldsymbol{\Theta}, \boldsymbol{Y}_{\bar{\Omega}} | \boldsymbol{X}, \boldsymbol{Y}_\Omega) = \frac{P(\boldsymbol{\Theta}, \boldsymbol{X}, \boldsymbol{Y})}{P(\boldsymbol{X}, \boldsymbol{Y}_\Omega)} \tag{12}$$

However, it is difficult to compute (12) because computing $P(\boldsymbol{X}, \boldsymbol{Y}_\Omega)$ requires marginalizing over all the parameters in $\boldsymbol{\Theta}$, which is often intractable.

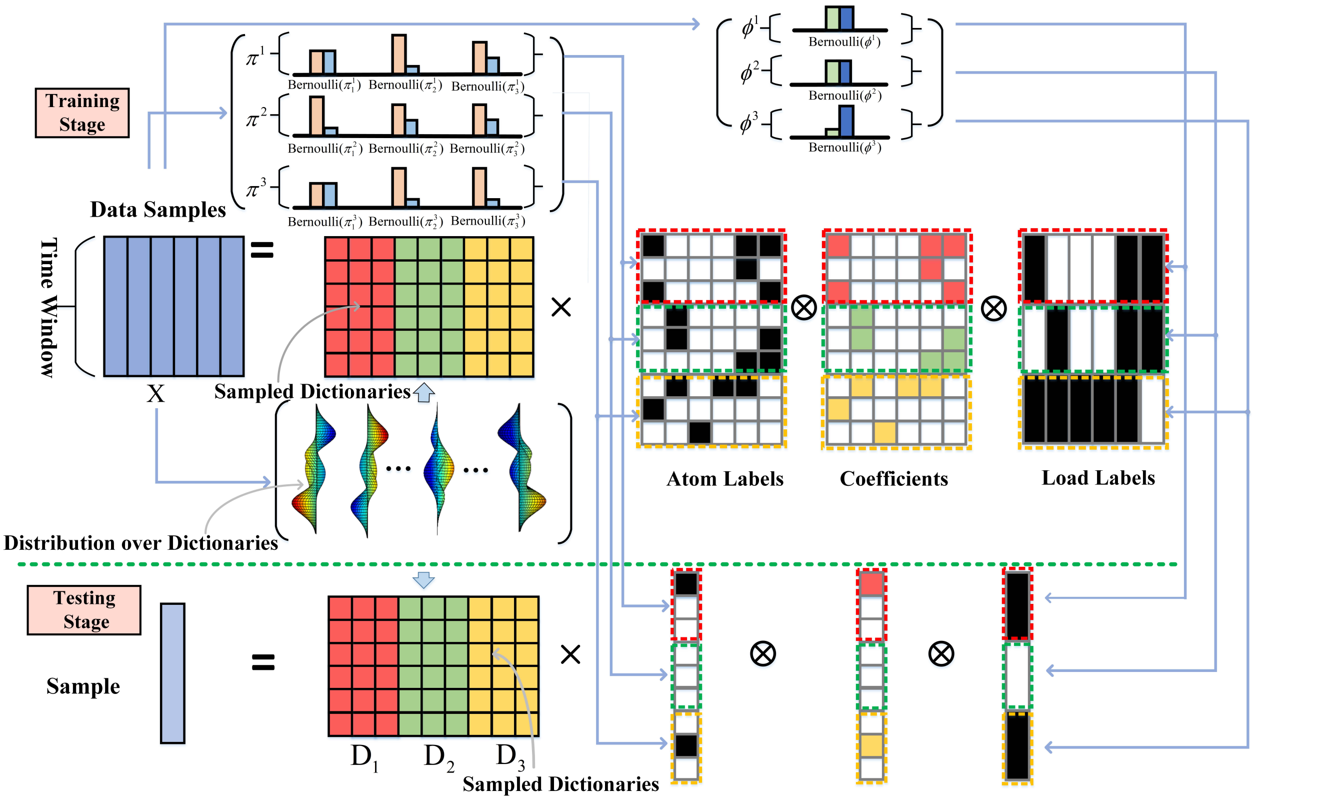


Fig. 1: An overall illustration of the proposed method. In the offline stage, the method learns the probability distributions of the dictionaries, the labels of existing dictionary atoms, the coefficients of the dictionary atoms, and the labels of the existing loads from the training data. In the online stage, based on the learned dictionary distributions, our method learns the probability distributions of the coefficients and labels for the testing sample.

Gibbs sampling [50], a standard Monte Carlo Markov Chain (MCMC) technique, is employed here to compute $P(\boldsymbol{\Theta}, \boldsymbol{Y}_{\bar{\Omega}}|\boldsymbol{X}, \boldsymbol{Y}_{\Omega})$. Gibbs sampling generates a Markov Chain of samples by sequentially sampling from the conditional distribution of one variable given all others, and the stationary distribution of the Markov Chain follows the desired joint distribution. Here, we sequentially sample from the conditional probability of one variable in $\boldsymbol{\Theta}$ and $\boldsymbol{Y}_{\bar{\Omega}}$ given all the other parameters in $\boldsymbol{\Theta}$, $\boldsymbol{Y}$, and $\boldsymbol{X}$. These conditional distributions can be computed explicitly because they have the known forms due to conjugate priors and are proportional to the joint distribution $P(\boldsymbol{\Theta}, \boldsymbol{X}, \boldsymbol{Y})$, which can be computed explicitly. Here, the conditional distribution of each variable given others are directly introduced. The detailed derivations are in the supplementary material. $p(\boldsymbol{d}_k^c|-)$ denotes the conditional distribution of $\boldsymbol{d}_k^c$ while keeping other variables fixed. The same rule applies to other notations.

(I) Sample $\boldsymbol{d}_k^c$ (for all $c = 1, ..., C$ and $k = 1, ..., K_c$), the kth dictionary of class c, from a Gaussian distribution with mean $\boldsymbol{\mu}_{\boldsymbol{d}_k^c}$ and covariance $\boldsymbol{\Sigma}_{\boldsymbol{d}_k^c}$, i.e.,

$$p(\boldsymbol{d}_k^c|-) \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{d}_k^c}, \boldsymbol{\Sigma}_{\boldsymbol{d}_k^c}) \tag{13}$$

where

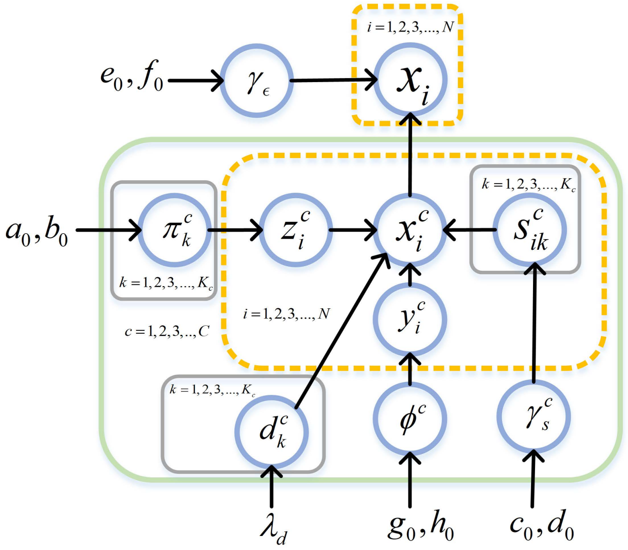


Fig. 2: Graphical representation of the proposed Bayesian dictionary learning model

$$\boldsymbol{\Sigma}_{\boldsymbol{d}_k^c} = (\lambda_d + \gamma_\epsilon \sum_{i=1}^{N} (z_{ik}^c s_{ik}^c y_i^c)^2)^{-1} \boldsymbol{I}_P \tag{14}$$

$$\boldsymbol{\mu}_{\boldsymbol{d}_k^c} = \gamma_\epsilon \boldsymbol{\Sigma}_{\boldsymbol{d}_k^c} \sum_{i=1}^{N} (z_{ik}^c s_{ik}^c y_i^c) \boldsymbol{x}_{i\boldsymbol{d}_k^c} \tag{15}$$

$$\text{and } \boldsymbol{x}_{i\boldsymbol{d}_k^c} = \boldsymbol{x}_i - \sum_{c=1}^{C} \boldsymbol{D}^c(\boldsymbol{z}_i^c \odot \boldsymbol{s}_i^c)y_i^c + \boldsymbol{d}_k^c(z_{ik}^c s_{ik}^c)y_i^c. \quad (16)$$

(II) Sample z_{ik}^c (for all $i = 1, ..., N$, $c = 1, ..., C$ and $k = 1, ..., K_c$) from a Bernoulli distribution

$$p(z_{ik}^c|-) \sim \text{Bernoulli}(\frac{\zeta_1}{\zeta_1 + 1 - \pi_k^c}) \quad (17)$$

where

$$\zeta_1 = \pi_k^c \text{exp}(-\frac{\gamma_\epsilon}{2}(-2\boldsymbol{x}_{i\boldsymbol{d}_k^c}^T \boldsymbol{d}_k^c(s_{ik}^c y_i^c) + \boldsymbol{d}_k^{c\,T} \boldsymbol{d}_k^c(s_{ik}^c y_i^c)^2)).$$

(III) Sample s_{ik}^c ($i = 1, ..., N$, $c = 1, ..., C$ and $k = 1, ..., K_c$) from a Gaussian distribution

$$p(s_{ik}^c|-) \sim \mathcal{N}(\boldsymbol{\mu}_{s_{ik}^c}, \boldsymbol{\Sigma}_{s_{ik}^c}) \quad (18)$$

where

$$\boldsymbol{\Sigma}_{s_{ik}^c} = (\gamma_s^c + \gamma_\epsilon(z_{ik}^c y_i^c)^2 ||\boldsymbol{d}_k^c||_2^2)^{-1} \quad (19)$$

$$\boldsymbol{\mu}_{s_{ik}^c} = \gamma_\epsilon \boldsymbol{\Sigma}_{s_{ik}^c}(z_{ik}^c y_i^c)(\boldsymbol{d}_k^c)^T \boldsymbol{x}_{i\boldsymbol{d}_k^c}. \quad (20)$$

(IV) Sample π_k^c ($c = 1, ..., C$ and $k = 1, ..., K_c$) from a Beta distribution

$$p(\pi_k^c|-) \sim \text{Beta}(\frac{a_0}{K_c} + \sum_{i=1}^{N} z_{ik}^c, \frac{b_0(K_c - 1)}{K_c} + N - \sum_{i=1}^{N} z_{ik}^c). \quad (21)$$

(V) Sample γ_s^c ($c = 1, ..., C$) from a Gamma distribution

$$p(\gamma_s^c|-) \sim \Gamma(\frac{NK_c}{2} + c_0, \frac{1}{2}\sum_{i=1}^{N} ||\boldsymbol{s}_i^c||_2^2 + d_0). \quad (22)$$

(VI) Sample γ_ϵ from a Gamma distribution

$$p(\gamma_\epsilon|-) \sim \Gamma(\frac{PN}{2} + e_0, \frac{1}{2}\sum_{i=1}^{N} ||\boldsymbol{x_i} - \sum_{c=1}^{C} \boldsymbol{D}^c(\boldsymbol{z}_i^c \odot \boldsymbol{s}_i^c)y_i^c||_2^2 + f_0). \quad (23)$$

(VII) Sample unknown labels y_i^c for (c, i) in $\bar{\Omega}$ drawn from a Bernoulli distribution

$$p(y_i^c|-) \sim \text{Bernoulli}(\frac{\zeta_2}{\zeta_2 + 1 - \phi^c}) \quad (24)$$

where

$$\zeta_2 = \phi^c \text{exp}(-\frac{\gamma_\epsilon}{2}(-2\boldsymbol{x}_{i\boldsymbol{D}^c}^T \boldsymbol{D}^c(\boldsymbol{z}_i^c \odot \boldsymbol{s}_i^c) + ||\boldsymbol{D}^c(\boldsymbol{z}_i^c \odot \boldsymbol{s}_i^c)||_2^2)) \quad (25)$$

$$\boldsymbol{x}_{i\boldsymbol{D}^c} = \boldsymbol{x}_i - \sum_{c=1}^{C} \boldsymbol{D}^c(\boldsymbol{z}_i^c \odot \boldsymbol{s}_i^c)y_i^c + \boldsymbol{D}^c(\boldsymbol{z}_i^c \odot \boldsymbol{s}_i^c)y_i^c. \quad (26)$$

(VIII) Sample ϕ^c from a Beta distribution

$$p(\phi^c|-) \sim \text{Beta}(g_0 + \sum_{i=1}^{N} y_i^c, h_0 + N - \sum_{i=1}^{N} y_i^c). \quad (27)$$

The details of the proposed method are shown in Algorithm 1 in the supplementary materials. The dictionary in each iteration is pruned to reduce the computational time and obtain a compact dictionary. Specifically, in each iteration, given c and k, the algorithm checks all the training data that are currently labeled as containing load c only. If z_{ik}^c are all zeros in all these training data i, the dictionary atom $\boldsymbol{d}_k^c$ is removed because it does not appear in any load c measurements. The corresponding π_k^c, z_{ik}^c, s_{ik}^c for all i are also removed.

Vector $\widetilde{\boldsymbol{x}}_i^t$ is the estimation of total power consumption of all loads in $\boldsymbol{x}_i$ in the t th iteration. Matrix $\widetilde{\boldsymbol{X}}^t$ includes all the $\bar{\boldsymbol{x}}_i^t$ for all i. The algorithm terminates if the change in $\widetilde{\boldsymbol{X}}^t$, measured by $\frac{\|\widetilde{\boldsymbol{X}}^t - \widetilde{\boldsymbol{X}}^{t-1}\|_F}{\|\widetilde{\boldsymbol{X}}^{t-1}\|_F}$, is less than pre-defined threshold ξ_1 or if the maximum number of iterations T_1 is achieved. ξ is a very small constant. T_1 can be set as a large value such that the Markov chain achieves the stationary distribution approximately.

For initialization, $\boldsymbol{D}^c$ is sampled randomly from data that contain label c. If no data contains label c, $\boldsymbol{D}^c$ is sampled randomly from the training data. $\boldsymbol{z}_i^c$ are initialized with all-one vectors. Every entry in $\boldsymbol{Y}_\Omega$ is initialized with 1. Given the initialization of the dictionaries $\boldsymbol{D}^c$, the sparse coefficients $\boldsymbol{S}$ are initialized by the solution to sparse coding [19] in (28).

$$\min_{\boldsymbol{S}} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{S}\|_2 + \lambda\|\boldsymbol{S}\|_1, \quad (28)$$

where λ is a regularization constant, and $\boldsymbol{D} = [\boldsymbol{D}^1, \boldsymbol{D}^2, ..., \boldsymbol{D}^C]$. The ℓ_1 norm measures the sum of the absolute values of all entries in a matrix and promotes a sparse solution. $\boldsymbol{S} = [\boldsymbol{S}^1; \boldsymbol{S}^2; ...; \boldsymbol{S}^C]$, and the ith column of $\boldsymbol{S}^c$ contains s_{ik}^c for all $k = 1, ..., K_c$. Algorithm 1 also handles naturally the case that all the labels are known . One only needs to skip lines 3-6 in Algorithm 1 about updating unknown y_i^c and ϕ^c, and everything else remains unchanged.

Our probabilistic model and the proposed learning method are different from conventional Bayesian dictionary learning such as [22]–[24]. Even though Ref. [24] considers learning C dictionaries using data from C classes, every training data point in [24] belongs to exactly one class, and the label is correctly known. Thus, it is still the conventional problem of learning each dictionary separately using the data in that class. The fundamental difference of our problem setup from conventional Bayesian dictionary learning is that every training sample $\boldsymbol{x}_i$ can contain up to C loads, and the load types that exist in $\boldsymbol{x}_i$ are not fully known. Thus, the new binary variables y_i^c are introduced in our model, and the probability distributions are updated using (24) and (27), and all the unknown y_i^c are sampled based on the currently estimated distribution in each iteration.

B. Online load disaggregation and uncertainty calculation

In real-time operations, given the aggregate data $\hat{\boldsymbol{x}}$, our approach estimates the consumption $\hat{\boldsymbol{x}}^c$ for load c using the learned probability distributions in the offline stage. A probabilistic model $\hat{\boldsymbol{x}}$ is described as follows. For all $c = 1, ..., C$, $k = 1, ..., K_c$

$$\hat{\boldsymbol{x}} = \sum_{c=1}^{C} \boldsymbol{D}^c(\hat{\boldsymbol{z}}^{\boldsymbol{c}} \odot \hat{\boldsymbol{s}}^{\boldsymbol{c}})\hat{y}^c + \hat{\boldsymbol{\epsilon}} \quad (29)$$

$$d_k^c \sim p(d_k^c|\boldsymbol{X}, \boldsymbol{Y}_\Omega) \quad (30)$$

$$\hat{\boldsymbol{x}}^c = \boldsymbol{D}^c(\hat{\boldsymbol{z}}^{\boldsymbol{c}} \odot \hat{\boldsymbol{s}}^{\boldsymbol{c}})\hat{y}^c + \frac{\hat{\boldsymbol{\epsilon}}}{C}. \tag{31}$$

$$\hat{\boldsymbol{z}}^c \sim \prod_{k=1}^{K_c} \text{Bernoulli}(\pi_k^c), \quad \pi_k^c \sim p(\pi_k^c|\boldsymbol{X}, \boldsymbol{Y}_\Omega) \tag{32}$$

$$\hat{\boldsymbol{s}}^c \sim \mathcal{N}(\boldsymbol{0}, \frac{1}{\gamma_s^c}\boldsymbol{I}_{K_c}), \quad \gamma_s^c \sim p(\gamma_s^c|\boldsymbol{X}, \boldsymbol{Y}_\Omega) \tag{33}$$

$$\hat{\boldsymbol{y}} \sim \prod_{c=1}^{C} \text{Bernoulli}(\phi^c), \ \phi^c \sim p(\phi^c|\boldsymbol{X}, \boldsymbol{Y}_\Omega) \tag{34}$$

$$\hat{\boldsymbol{\epsilon}} \sim \mathcal{N}(\boldsymbol{0}, \frac{1}{\hat{\gamma}_\epsilon}\boldsymbol{I}_P) \tag{35}$$

$$\hat{\gamma}_\epsilon \sim \Gamma(e_1, f_1) \tag{36}$$

Equations (29)-(36) are similar to (1) to (11) with two differences. First, the learned probability distributions of $p(d_k^c|\boldsymbol{X}, \boldsymbol{Y})$, $p(\pi_k^c|\boldsymbol{X}, \boldsymbol{Y}_\Omega)$, $p(\gamma_s^c|\boldsymbol{X}, \boldsymbol{Y}_\Omega)$, $p(\phi^c|\boldsymbol{X}, \boldsymbol{Y}_\Omega)$, as shown in (30) to (34), from the offline stage are directly employed and not updated in the online stage. $p(\gamma_\epsilon|\boldsymbol{X}, \boldsymbol{Y}_\Omega)$ is sampled once to initialize $\hat{\gamma}_\epsilon$. Second, the pre-determined parameters e_1 and f_1 in the Gamma distribution for $\hat{\gamma}_\epsilon$ are different from the counterparts in the offline stage. That is because in the offline stage the method learns the average noise distribution of all N samples, while in the testing stage, the method focuses on the noise distribution of each individual testing sample.

Given $\hat{\boldsymbol{x}}$, the posterior distributions of $\hat{y}^c$, $\hat{z}_k^c$, $\hat{s}_k^c$, and $\hat{\gamma}_\epsilon$ are computed using Gibbs sampling. The updating rules are similar to those in Algorithm 1 with minor changes. Specifically, one only needs to replace $\boldsymbol{x}_i$, y_i^c, z_{ik}^c, s_{ik}^c, γ_ϵ for the training data with $\hat{\boldsymbol{x}}$, $\hat{y}^c$, $\hat{z}_k^c$, $\hat{s}_k^c$, $\hat{\gamma}_\epsilon$ for the testing data in equations (17), (18), (24). Moreover, the updating rule for $\hat{\gamma}_\epsilon$ is

$$p(\hat{\gamma}_\epsilon|-) \sim \Gamma(\frac{P}{2}+e_1, \frac{1}{2}||\hat{\boldsymbol{x}} - \sum_{c=1}^{C} \boldsymbol{D}^c(\boldsymbol{z}_i^c \odot \boldsymbol{s}_i^c)y^c||_2^2 + f_1) \tag{37}$$

The algorithm details are summarized in Algorithm 2 in the supplementary materials.

After computing all the posterior distributions, one can estimate the distribution of $\hat{x}^c$ from (31). Then use the mean as the estimate of the consumption of load c and use the covariance to estimate the uncertainty. Because it is intractable to obtain the closed-form expression of the probability distribution of $\hat{x}^c$, Monte-Carlo integration [51] is employed to compute the mean and variance approximately.

To simplify the notations, let $\boldsymbol{\Psi} = \{\boldsymbol{D}^c, \hat{\boldsymbol{z}}^c, \hat{\boldsymbol{s}}^c, \hat{y}^c, \hat{\gamma}_\epsilon\}$ denote the set of variables related to computing $\hat{x}^c$, and define

$$f(\boldsymbol{\Psi}) = \boldsymbol{D}^c(\hat{\boldsymbol{z}}^{\boldsymbol{c}} \odot \hat{\boldsymbol{s}}^{\boldsymbol{c}})\hat{y}^c \tag{38}$$

The predictive mean can be approximated by

$$E[\hat{\boldsymbol{x}}^c] \approx \frac{1}{L}\sum_{l=1}^{l=L} f(\boldsymbol{\Psi}^l) \tag{39}$$

where each $\boldsymbol{\Psi}^l$ is independently drawn from the learned probabilities distributions of $\boldsymbol{D}^c$, $\hat{\boldsymbol{z}}^c$, $\hat{\boldsymbol{s}}^c$, and $\hat{y}^c$. When the number of Monte-Carlo samples L is sufficiently large, the right-hand side of (39) provides a rather accurate estimate of the mean of $\hat{\boldsymbol{x}}^c$, which is used as an estimate of the load c consumption. Similarly, the predictive covariance can be computed by

$$\begin{aligned}\text{Var}[\hat{\boldsymbol{x}}^c] =& E[\hat{\boldsymbol{x}}^c\hat{\boldsymbol{x}}^{cT}] - E[\hat{\boldsymbol{x}}^c]E[\hat{\boldsymbol{x}}^c]^T \\ \approx& \frac{\boldsymbol{I}_P}{LC}\sum_{l=1}^{l=L}\frac{1}{\hat{\gamma}_\epsilon^l} + \frac{1}{L}\sum_{l=1}^{l=L} f(\boldsymbol{\Psi}_l)f(\boldsymbol{\Psi}^l)^T \\ &-(\frac{1}{L}\sum_{l=1}^{l=L} f(\boldsymbol{\Psi}^l))(\frac{1}{L}\sum_{l=1}^{l=L} f(\boldsymbol{\Psi}^l)^T\end{aligned} \tag{40}$$

Let σ_i $(i = 1, ..., P)$ be all the singular values of Var$[\hat{x}^c]$. An uncertainty index U_c for each load c and the uncertainty index U_{all} for all output loads are defined as

$$U_c = \Sigma_{i=1}^P \sigma_i \tag{41}$$

$$U_{\text{all}} = \Sigma_{c=1}^C U_c \tag{42}$$

Intuitively, if a random variable has a large variance, then the estimation of its realization may have high uncertainty. The uncertainty indices in (41) and (42) can characterize this phenomenon.

C. The influence of parameter selections

The proposed hierarchical model requires several hyper-parameters. λ_d in (3) is a constant and can be set as the length of the time window. Four pairs $(a_0, b_0), (c_0, d_0), (e_0, f_0), (g_0, h_0)$ of parameters are used in the prior distributions (5), (8), (9) and (11). As shown in [22], c_0 and d_0 are set as very small values (10^{-6} both in [22] and here), and they are non-informative priors in Gamma distributions, providing little information to the experiments.

Regarding the other three pairs, fixing either one of the pair and tuning another has similar performance. For example, fixing a_0 and increasing b_0 have similar performance with fixing b_0 and decreasing a_0. A larger b_0 leads to a smaller mean of the prior distribution of π_k^c, which in turn leads to more zero entries in $\boldsymbol{z}_i^c$. Similarly, g_0 and h_0 affect the number of zeros in $\boldsymbol{Y}_{\bar{\Omega}}$. When g_0 is fixed, a larger h_0 leads to a smaller ϕ^c and, correspondingly, more zeros in $\boldsymbol{Y}_{\bar{\Omega}}$. Increasing f_0 while fixing e_0 leads to a smaller γ_ϵ, which in turn leads to a larger variance of the measurement noise ϵ. Because these parameters affect prior distributions only and are independent of the data, they have limited impact on the learned posterior distributions. As one can see in Section IV-C, the method has very similar disaggregation performance in a wide range of parameter selections.

One remark is that our method is robust to the initial dictionary size K_c because it prunes the dictionaries in computing the posterior distribution. This is one of the advantages over the deterministic approach in [19], which requires accurate estimation of the dictionary size.

IV. Numerical Experiments

A. Experimental Setup

1) Datasets: The numerical evaluations are performed on two datasets of partially labeled aggregate data. The first one, referred to as the "Ind-Ind-Solar" dataset, is the same as that in [19]. It contains three types of loads, two industrial sites and one solar generation. The industrial loads are from EnerNOC GreenButton Data [52], and the solar generation is from National Renewable Energy Laboratory (NREL) [53]. Each measurement has an 8-hour time window from 8:00 am to 4:00 pm with a 5-minute resolution. The power consumption of load 1 (industry load from one site) varies from 23 MW to 73 MW, and load 2 (industry load from the other site) varies from 39 MW to 74 MW. The power generation of load 3 (solar generation) is from 4 MW and 65 MW. All the measurements are denoised by a Gaussian filter and then added together to generate 360 training samples and 300 testing samples. Each training sample is labeled with one load, although it may contain up to all three loads. No label is provided for the testing data. γ denotes the percentage of measurements of individual loads in the training data. $\gamma = 70\%$ means that 70% of the training data labeled as c ($c = 1, 2, 3$) contain load c only and the 30% of data contain other loads.

The second dataset, referred to as the "Resi-Ind-Solar" dataset, is constructed by replacing the first industrial load in the previous dataset with a residential load. The residential load is from Pecan Street[2] and contains 25 homes in Austin. The measurement has an 8-hour time window from 8:00 am to 4:00 pm with a 15-minute resolution. The power consumption of residential load varies from 12 MW to 95 MW. We keep the same time window and down-sample the industrial and solar loads to the 15-minute resolution. γ is set as 70%.

2) Methods: Our proposed Bayesian energy disaggregation method at the substation (abbreviated by "B-EDS") is compared with the deterministic EDS method in [19] (abbreviated by "D-EDS"). Other dictionary-learning-based disaggregation methods such as [11] and [45], as well as deep-learning-based methods [13], [38], [39] need training data of individual loads to learn each dictionary. When the measurements contain multiple types of loads and are not fully labeled, measurements labeled as load c can in fact contain other loads in the EDS problem, then the learned dictionary of each load often contains patterns of other loads, resulting in significant disaggregation errors. This limitation of these methods has already been demonstrated empirically in [19], see Table III to Table V in [19] for details. Therefore, in this paper, B-EDS is compared with D-EDS, which is the state-of-the-art method for EDS with partially labeled aggregate data.

In B-EDS method, Algorithm 1 is employed to learn the patterns from recorded data. Algorithm 2 is implemented online to disaggregate real-time measurements and compute the uncertainty index. If not otherwise specified, the parameters are set as follows: $a_0 = 1$, $b_0 = 10^4$, $c_0 = 10^{-6}$, $d_0 = 10^{-6}$, $e_0 = 10^{-3}$, $f_0 = 10$, $g_0 = 1$, $h_0 = 10^5$, $e_1 = 10^{-6}$, $f_1 = 0.06$, $\lambda_d = 96, \xi_1 = 0.01, \xi_2 = 0.001$. The robustness of our method to the parameter selection is demonstrated in Section IV-B. The initial values of K_c (for all c), π_k^c, ϕ^c, γ_s^c, γ_ϵ are set as 10, 0.01, 0.01, 1, 100, respectively. λ in (28) is set as 0.001 to initialize Algorithm 1 and $0.01 - 0.1$ to initialize Algorithm 2. T_1 is 5000, and T_2 is 2000. $L = 50$ in (39) and (40). All the following results for D-EDS are averaged over 50 times for a fair comparison.

All the experiments are run in MATLAB 2019 on a desktop with 3.1 GHz Intel Core i9 and 32 GB memory. The training time for B-EDS is around 50 seconds, and the testing time for each testing sample is around 4 seconds.

3) Performance evaluation: Several metrics are employed to measure the disaggregation accuracy. Root Mean Square Error (RMSE) [45] is a standard metric to measure the disaggregation error. RMSE$_c$ measures the average error of disaggregating load c from M aggregate measurements. $\hat{x}_i^c$ and $\bar{x}_i^c$ in $\mathbb{R}^P$ represent the estimation and the ground-truth consumption of load c in the ith sample, respectively.

$$\text{RMSE}_c = \sqrt{\frac{\Sigma_{i=1}^M \|\hat{x}_i^c - \bar{x}_i^c\|_2^2}{PM}}. \tag{43}$$

A new metric, weighted root mean square error (WRMSE), is proposed here to measure the weighted average disaggregation error, as shown in (44).

$$\text{WRMSE}_c = \sqrt{\frac{\Sigma_{i=1}^M \frac{\|\hat{x}_i^c - \bar{x}_i^c\|_2^2}{U_c(\hat{x}_i^c)}}{P\Sigma_{i=1}^M \frac{1}{U_c(\hat{x}_i^c)}}} \tag{44}$$

Compared with RMSE, an additional weight $1/U_c(\hat{x}_i^c)$ is multiplied to the estimation error of load c consumption in sample i, where $U_c(\hat{x}_i^c)$ is the uncertainty index of the estimation $\hat{x}_i^c$. A larger uncertainty index indicates that the estimation is less reliable. Therefore, the corresponding error is multiplied with a smaller weight when computing the overall error in WRMSE. From its definition in (44), if the uncertain index is the same for all the training samples, WRMSE is the same as RMSE. If the estimations that are less accurate have larger uncertainty indices, then WRMSE can be much smaller than RMSE. In fact, WRMSE being much less than RMSE indicates that the unreliable estimation can be identified by the uncertain index. To see this, consider a simple case that the individual reconstructed errors for three samples $||\hat{x}_i^c - \bar{x}_i^c||_2, i = 1, 2, 3$ are 80, 60 and 2, respectively. Suppose the corresponding uncertainty indices are 64, 36, 1, respectively and $P = 96$. Then RMSE is 5.90 while the WRMSE is 1.43. That is because the first sample has a high estimation error but its uncertainty index is also high.

The Total-Error-Rate (TER) [19] is employed to measure the total disaggregation error of all the loads,

$$\text{TER} = \frac{\Sigma_{i=1}^M \Sigma_{c=1}^C \min(\|\hat{x}_i^c - \bar{x}_i^c\|_1, \|\bar{x}_i^c\|_1)}{\Sigma_{i=1}^M \Sigma_{c=1}^C \|\bar{x}_i^c\|_1} \times 100\% \tag{45}$$

TER belongs to $[0, 1]$, and a small TER corresponds to a small disaggregation error. Our B-EDS method also estimates the labels in the testing data. The label estimation for a testing sample is called successful if the existence of every load in that

[2] http://www.pecanstreet.org/

sample is correctly labeled. The average classification accuracy (CA) of M testing samples is measured by

$$\text{CA} = \frac{\text{Number of successful predictions}}{M} \times 100\%. \tag{46}$$

B. Disaggregation Performance

1) Performance on denoised measurements: Table I compares the disaggregation performance of B-EDS and D-EDS on "Ind-Ind-Solar" dataset when the measurements are denoised before implementing the methods. This is the setup considered in [19] and is used as a baseline here. Because D-EDS does not return the load labels and uncertain indices, its classification accuracy and WRMSE are not reported. Both methods perform better when γ is larger. B-EDS method has a higher disaggregation accuracy than D-EDS even when the data are deonised. One important observation is that the WRMSE of each load by B-EDS is significantly smaller than the corresponding RMSE. As discussed after (44), this indicates that those inaccurate estimates are accompanied by large uncertainty indices, and one can indeed use the uncertainty index to evaluate the reliability of the estimation. Although the RMSE of B-EDS is only slightly better than that of D-EDS on denoised data, the WRMSE of B-EDS is much smaller than its RMSE. That shows B-EDS can differentiate reliable and unreliable estimates, and the average error of those reliable estimates, i.e., with small uncertainty indices, are much smaller than the average error of all the estimates.

Table I: Disaggregation performance of B-EDS and D-EDS on "Ind-Ind-Solar" dataset with denoised data

	B-EDS		D-EDS	
	$\gamma = 70\%$	$\gamma = 50\%$	$\gamma = 70\%$	$\gamma = 50\%$
RMSE$_1$	5.89	7.03	6.63	7.43
RMSE$_2$	5.14	5.29	5.12	6.12
RMSE$_3$	5.67	6.54	6.11	6.31
WRMSE$_1$	0.16	0.23	-	-
WRMSE$_2$	0.13	0.17	-	-
WRMSE$_3$	0.13	0.22	-	-
TER	8.69%	10.89%	9.91%	11.63%
CA	95.00%	92.67%	-	-

2) Performance on noisy measurements: These two methods are also compared when the data are noisy. Gaussian noise i.i.d. drawn from $\mathcal{N}(0, \sigma^2)$ are added to every denoised aggregate measurement in both the training and testing samples. Here $b_0 = 1$ and other parameters are the same as in Section IV-A. Table II shows that B-EDS is much more robust to noise than D-EDS. On denoised data, the TER of B-EDS is 1% better than that of D-EDS (Table I), corresponding to about 10% performance improvement. When σ is between 1 and 3, the TER of B-EDS is 2.5-3% lower than that of D-EDS, which corresponds to about 25% performance improvement. That indicates B-EDS handles noise much better. Both methods are also evaluated on the original measurements before denoising. B-EDS again performs much better than D-EDS. Moreover, WRMSE of B-EDS is significantly smaller than the corresponding RMSE.

The dictionary size in D-EDS is selected to achieve a predefined approximation threshold of the training data. Therefore, the dictionary sizes K_1 and K_2 and K_3 increase significantly when the noise level increases. For a fair comparison, the initial dictionary sizes in B-EDS is set the same as those in D-EDS. Because B-EDS models the noise directly and can prune the dictionary accordingly, one can see that the final dictionary sizes are much smaller than that of D-EDS and consistent under different noise levels.

Table II: Disaggregation performance of B-EDS and D-EDS on "Ind-Ind-Solar" dataset when data contain noises ($\gamma = 70\%$)

	B-EDS			D-EDS		
	$\sigma = 1$	$\sigma = 3$	original	$\sigma = 1$	$\sigma = 3$	original
RMSE$_1$	7.06	7.08	7.09	9.00	11.03	10.81
RMSE$_2$	5.58	6.18	6.24	4.88	6.17	5.79
RMSE$_3$	6.50	5.99	7.20	9.30	8.51	10.48
WRMSE$_1$	3.06	3.39	3.59	-	-	-
WRMSE$_2$	0.14	0.16	0.11	-	-	-
WRMSE$_3$	0.16	0.11	0.14	-	-	-
TER	10.17%	10.99%	12.44%	12.66%	14.08%	15.18%
CA	94.67%	95.67%	94.67%	-	-	-
K_1	2	2	2	13	44	22
K_2	3	2	3	10	42	46
K_3	2	3	2	16	46	24

3) Impact of label errors: The disaggregation performance of B-EDS with partial label errors is evaluated on the denoised "Ind-Ind-Solar" dataset. $\gamma = 70\%$. ρ denotes the percentage of erroneous labels among all available labels. The erroneous patterns include setting the labels of pure load 1 measurements as "load 2," setting the labels of pure load 2 measurements as "load 3," and setting the labels of pure load 3 measurements as "load 1." The results without label errors are repeated in the first column of Table III to compare. One can see that B-EDS is not sensitive to label errors. The average performance such as RMSE, TER, and CA degrade slightly when ρ increases. An interesting observation is that WRMSE always stays small and does not change much when ρ changes. That means although a large percentage of wrong labels can lead to errors in the estimation of some loads, those inaccurate estimates are accompanied by high uncertain indices and can thus be identified by the operator.

Table III: The disaggregation results of B-EDS with label errors on "Ind-Ind-Solar" dataset ($\gamma = 70\%$, $\sigma = 0$)

ρ	0	5%	10%	15%
RMSE$_1$	5.89	8.60	7.92	8.16
RMSE$_2$	5.14	5.60	5.14	6.46
RMSE$_3$	5.67	7.26	7.47	7.28
WRMSE$_1$	0.16	0.27	0.21	0.21
WRMSE$_2$	0.13	0.14	0.14	0.15
WRMSE$_3$	0.13	0.16	0.15	0.16
TER	8.69%	10.90%	11.40%	12.01%
CA	95.00%	91.33%	92.33%	91.67%

4) Performance on the "Resi-Ind-Solar" dataset.: Table IV compares the disaggregation performance of B-EDS and D-EDS on the "Resi-Ind-Solar" dataset $\gamma = 70\%$. All the other parameters are kept the same except that $b_0 = 1$, $f_0 = 1$ and $f_1 = 0.003$. Similar to the performance on "Ind-Ind-Solar" dataset, the TER of B-EDS is around 1% higher than that of D-EDS on denoised data, corresponding to 8% performance improvement. When the measurements are noisy, B-EDS has a much higher disaggregation accuracy than D-EDS: TER of B-EDS is 3% smaller than that of D-EDS, corresponding to 25%

performance improvement. Moreover, the WRMSE for the residential load by B-EDS is higher than those for industrial and solar loads. That is because the load patterns of residential loads are more volatile than others, leading to less accuracy than other loads.

Table IV: Disaggregation performance of B-EDS and D-EDS on "Resi-Ind-Solar" dataset with different noise level ($\gamma = 70\%$)

	B-EDS			D-EDS		
	$\sigma = 0$	$\sigma = 1$	$\sigma = 3$	$\sigma = 0$	$\sigma = 1$	$\sigma = 3$
RMSE$_1$	9.03	9.88	10.17	8.25	9.34	11.39
RMSE$_2$	8.74	8.51	8.74	9.14	9.54	8.53
RMSE$_3$	5.41	7.06	6.37	6.70	9.88	11.97
WRMSE$_1$	6.54	7.30	9.32	-	-	-
WRMSE$_2$	0.12	0.14	0.13	-	-	-
WRMSE$_3$	0.05	0.07	0.07	-	-	-
TER	14.07%	15.42%	16.72%	15.28%	18.28%	19.94%
CA	93.00%	92.00%	91.33%	-	-	-
K_1	3	3	3	4	5	16
K_2	3	3	3	4	5	15
K_3	4	3	3	4	6	17

C. The influence of parameter selections

Table V: The influence of b_0 (a_0 is fixed and $a_0 = 1$)

b_0	1	10	10^2	10^3	10^4	10^5
TER	12.35%	10.29%	11.08%	11.42%	8.69%	10.80%
CA	92.33%	94.33%	93.00%	92.33%	95.00%	92.33%

The impact of the hyper parameters in the prior distributions on the performance of B-EDS are studied numerically. Following the discussion in Section III.C, we mainly focus on the three pairs $(a_0, b_0), (e_0, f_0), (g_0, h_0)$, and fix one while varying the other in each pair. Table V shows the impact of varying b_0 while fixing other parameters. One can see that B-EDS achieves low disaggregation errors with a wide range of b_0. Therefore, the algorithm is not sensitive to the selection of b_0. Table VI demonstrates that increasing the h_0 improves the classification accuracy and h_0 can be selected from a wide range. Table VII shows the impact of f_0 while other parameters are fixed. One can see that B-EDS is also not sensitive to the selection of f_0.

Table VI: The influence of h_0 (g_0 is fixed and $g_0 = 1$)

h_0	10^4	10^5	10^6	10^7
TER	11.67%	8.69%	10.71%	11.47%
CA	88.33%	95.00%	95.00%	91.67%

Table VII: The influence of f_0, (e_0 is fixed and $e_0 = 10^{-3}$)

f_0	1	5	10	15	20	25
TER	9.74%	9.80%	8.69%	10.13%	11.75%	11.60%
CA	94.33%	94.33%	95.00%	93.67%	91.33%	92.00%

B-EDS is robust to the initial dictionary size and prunes the dictionaries in computing the posterior distribution. In Table VIII, let K_c denote the initial dictionary sizes of all three loads and vary K_c. One can see that the final dictionary sizes are much smaller than the initial values due to the pruning. Moreover, the disaggregation and classification performance are consistent even though there are minor differences in the final dictionary sizes.

Table VIII: The influence of initial dictionary size

Initial K_c	6	10	14	18	22	26
TER	10.92%	8.69%	9.41%	11.05%	10.23%	10.31%
CA	92.00%	95.00%	92.67%	92.33%	95.33%	92.67%
K_1	3	5	7	6	8	10
K_2	4	7	9	8	8	10
K_3	4	6	6	8	8	9

D. The uncertainty index to measure the reliability of the results

Table IX: The uncertainty index and the disaggregation performance with different cases

	Case 1	Case 2	Case 3	Case 4	Case 5
U$_1$	182.17	143.89	318.89	0.13	129.35
U$_2$	294.26	331.58	129.25	0.13	616.57
U$_3$	196.08	272.35	408.42	943.39	705.54
U$_{\text{all}}$	673.52	748.81	858.56	971.02	1451.47
B-EDS TER	2.86%	4.37%	5.29%	15.95%	12.22%
D-EDS TER	6.56%	8.59%	9.01%	20.26%	18.02%

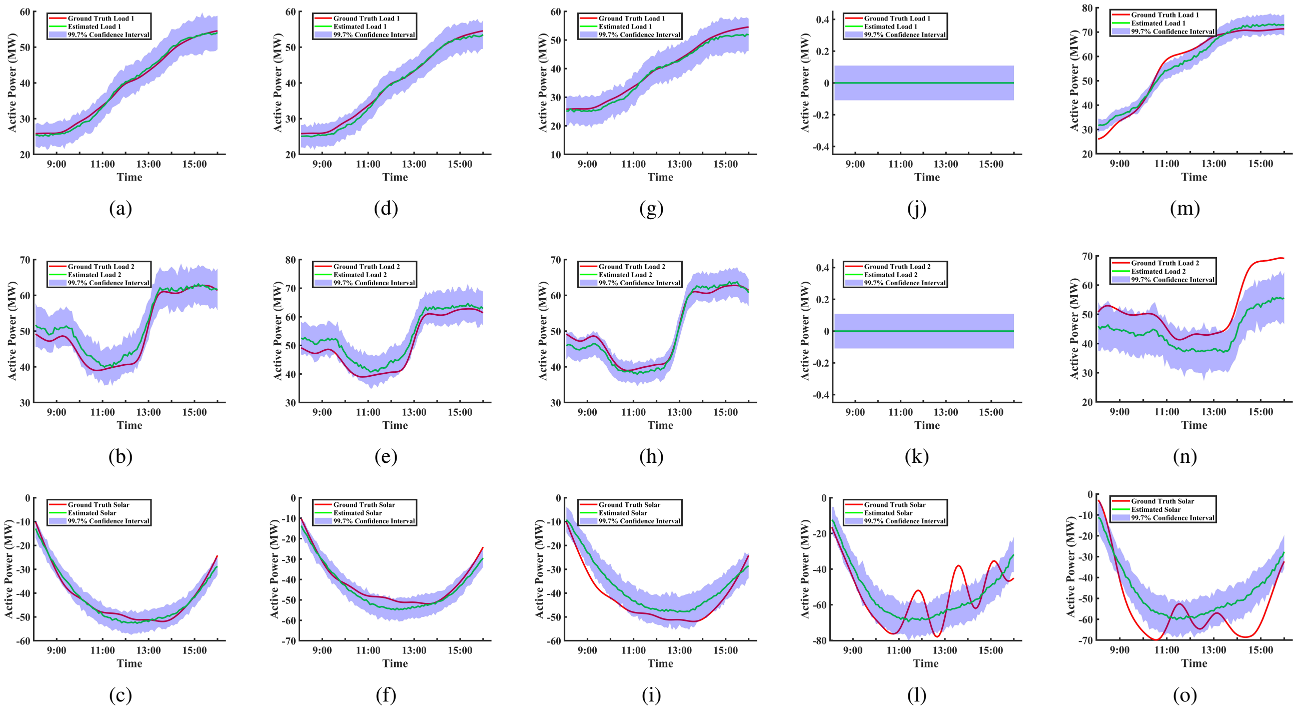
One main advantage of B-EDS over D-EDS is that B-EDS provides the uncertainty index of the returned disaggregation result, and the users can use the uncertainty index to evaluate the reliability of the results. Five case studies are provided here to demonstrate the effectiveness of uncertainty index.

- Case 1 contains three loads and is randomly selected from the testing samples used in Table I. It is used as a baseline case.
- Case 2 uses the same data as Case 1 but adds an i.i.d. Gaussian noise from $N(0, 3^2)$ to every data point.
- Case 3 adds an i.i.d. noise from $\mathcal{N}(0, 5^2)$ to every data point of the data in Case 1.
- Case 4 contains one solar load only, but its pattern is different from the solar patterns in the training data.
- Case 5 contains three loads, and the solar load has a different pattern from the training data.

Fig. 4 visualizes the different patterns of solar generation in the training and testing data for cases 4 and 5.

Fig. 3 shows the disaggregation performance of these five cases. The ground-truth measurements, the predictive mean of the estimation, and the 99.7% confidence interval of the estimation for each load are plotted. At each time instant, the two ends of the confidence interval are the mean value minus and plus three times the square root of the corresponding diagonal entry of the covariance matrix. Note that the confidence interval is for visualization only and does not characterize the correlations among measurements at different time instants. One shall use the uncertainty index for rigorous characterization of the uncertainty. One can see that in Cases 1-3, the ground-truth consumption is mostly within the confidence level, while as the noise level increases, the estimation error increases slightly (see Fig. 3(i)). In Cases 4 and 5, because the solar pattern is different from those in the training data, the estimation error is much larger, and the ground-truth solar generation deviates from the estimation.

Table IX lists the uncertainty indices and the disaggregation error for these five cases. From Cases 1-3, one can see that when the noise level in the measurements increases, the



(a) (d) (g) (j) (m)

(b) (e) (h) (k) (n)

(c) (f) (i) (l) (o)

Fig. 3: The load disaggregation performance of five cases in Table IX. (a)-(c) show the ground-truth data and the disaggregation results of loads 1, 2, and 3 respectively for Case 1. (d)-(f) show the results for Case 2 where the test data contain i.i.d. Gaussian noise $\mathcal{N}(0, 3^2)$. (g)-(i) show Case 2 where the test data contain i.i.d. Gaussian noise $\mathcal{N}(0, 5^2)$. (j)-(l) show Case 4 which contains one solar load with a different pattern from the training data. (m)-(o) show Case 5 that contains three loads, and the solar load has a different pattern from the training data.

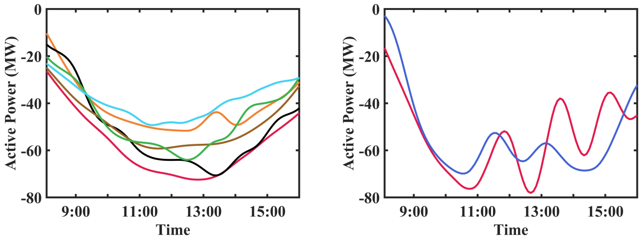


Fig. 4: The typical solar patterns of training data and some volatile testing samples. Left: typical patterns of training data. Right: typical patterns of volatile testing samples

uncertainty level increases slightly. From the uncertainty index for the solar generation (U_3), one can see that the index is much higher in Cases 4 and 5 when the pattern is different from the patterns in the training data. In Case 4, B-EDS can correctly identify that loads 1 and 2 do not exist and the uncertainty indices for these two loads are small. Due to the existence of the other two loads in Case 5, the disaggregation accuracy of the other two loads is also negatively affected, and thus the overall uncertainty U_{all} is higher than that in Case 4. Therefore, the uncertainty index can be used as a measure of the reliability of the disaggregation results. Table IX also compares the TER of B-EDS and D-EDS, and the former performs much better than the latter.

V. Conclusions

This paper develops a Bayesian-dictionary-learning-based load disaggregation approach for aggregate measurements with partial labels at the substation. The proposed approach improves the disaggregation performance, especially when the measurements contain noisy. An uncertainty index is provided about the returned estimation. This index can be used for the operator to evaluate the reliability of the disaggregation results and is especially useful in the presence of volatile loads and renewable generations. No previous works on EDS provide an uncertainty measure. The new error metric WRMSE verifies that the unreliable estimation can be identified by the uncertainty index.

One future direction is to connect side information such as weather and industrial distributions with the data-driven method. Another direction is to evaluate the disaggregation performance of other volatile loads such as electrical vehicles and wind power. We will also investigate the disaggregation method when the given label information contain errors.

References

[1] M. S. S. Abad, J. Ma, D. Zhang, A. S. Ahmadyar, and H. Marzooghi, “Probabilistic assessment of hosting capacity in radial distribution systems,” *IEEE Trans. Sustain. Energy*, vol. 9, no. 4, pp. 1935–1947, 2018.

[2] S. Wang, Y. Dong, L. Wu, and B. Yan, “Interval overvoltage risk based pv hosting capacity evaluation considering pv and load uncertainties,” *IEEE Trans. Smart Grid*, vol. 11, no. 3, pp. 2709–2721, 2019.

[3] B. Chen, C. Chen, J. Wang, and K. L. Butler-Purry, “Sequential service restoration for unbalanced distribution systems and microgrids,” *IEEE Trans. Power Syst.*, vol. 33, no. 2, pp. 1507–1520, 2018.

[4] Y. Wang, Y. Xu, J. He, C.-C. Liu, K. P. Schneider, M. Hong, and D. T. Ton, "Coordinating multiple sources for service restoration to enhance resilience of distribution systems," *IEEE Trans. Smart Grid*, vol. 10, no. 5, pp. 5781–5793, 2019.

[5] 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 Trans. Power Syst.*, vol. 33, no. 3, pp. 3255–3264, 2018.

[6] M. Sun, T. Zhang, Y. Wang, G. Strbac, and C. Kang, "Using bayesian deep learning to capture uncertainty for residential net load forecasting," *IEEE Trans. Power Syst.*, vol. 35, no. 1, pp. 188–201, 2019.

[7] N. Mahdavi and J. H. Braslavsky, "Modelling and control of ensembles of variable-speed air conditioning loads for demand response," *IEEE Trans. Smart Grid*, vol. 11, no. 5, pp. 4249–4260, 2020.

[8] Z. Xuan, X. Gao, K. Li, F. Wang, X. Ge, and Y. Hou, "Pv-load decoupling based demand response baseline load estimation approach for residential customer with distributed pv system," *IEEE Trans. Ind. Appl.*, vol. 56, no. 6, pp. 6128–6137, 2020.

[9] R. A. Jabr, "Robust volt/var control with photovoltaics," *IEEE Trans. Power Syst.*, vol. 34, no. 3, pp. 2401–2408, 2019.

[10] X. Zhou, M. Farivar, Z. Liu, L. Chen, and S. H. Low, "Reverse and forward engineering of local voltage control in distribution networks," *IEEE Trans. Automat. Contr.*, vol. 66, no. 3, pp. 1116–1128, 2021.

[11] J. Z. Kolter, S. Batra, and A. Y. Ng, "Energy disaggregation via discriminative sparse coding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 1153–1161.

[12] W. Kong, Z. Y. Dong, J. Ma, D. J. Hill, J. Zhao, and F. Luo, "An extensible approach for non-intrusive load disaggregation with smart meter data," *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3362–3372, 2016.

[13] K. Chen, Y. Zhang, Q. Wang, J. Hu, H. Fan, and J. He, "Scale-and context-aware convolutional non-intrusive load monitoring," *IEEE Trans. Power Syst.*, 2019.

[14] K. He, L. Stankovic, J. Liao, and V. Stankovic, "Non-intrusive load disaggregation using graph signal processing," *IEEE Trans. Smart Grid*, vol. 9, no. 3, pp. 1739–1747, 2016.

[15] G. W. Hart, "Nonintrusive appliance load monitoring," *Proceedings of the IEEE*, vol. 80, no. 12, pp. 1870–1891, 1992.

[16] J. M. Gillis, S. M. Alshareef, and W. G. Morsi, "Nonintrusive load monitoring using wavelet design and machine learning," *IEEE Trans. Smart Grid*, vol. 7, no. 1, pp. 320–328, 2015.

[17] S. M. Tabatabaei, S. Dick, and W. Xu, "Toward non-intrusive load monitoring via multi-label classification," *IEEE Trans. Smart Grid*, vol. 8, no. 1, pp. 26–40, 2016.

[18] M. Tabone, S. Kiliccote, and E. C. Kara, "Disaggregating solar generation behind individual meters in real time," in *Proc. 5th Conf. Systems for Built Environments*. ACM, 2018, pp. 43–52.

[19] 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 Trans. Power Syst.*, p. Early Access, 2020.

[20] W. Gao, L. Wang, Z.-H. Zhou *et al.*, "Risk minimization in the presence of label noise," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016.

[21] O. Dekel and O. Shamir, "Good learners for evil teachers," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 233–240.

[22] M. Zhou, H. Chen, J. Paisley, L. Ren, G. Sapiro, and L. Carin, "Non-parametric bayesian dictionary learning for sparse image representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 2295–2303.

[23] J. Paisley and L. Carin, "Nonparametric factor analysis with beta process priors," in *Proc. 29th Int. Conf. Mach. Learn.*, 2009, pp. 777–784.

[24] N. Akhtar, F. Shafait, and A. Mian, "Discriminative bayesian dictionary learning for classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 12, pp. 2374–2388, 2016.

[25] D. Chen and D. Irwin, "Sundance: Black-box behind-the-meter solar disaggregation," in *Proc. 18th International Conf. Future Energy Systems*, ser. e-Energy '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 45–55.

[26] P. Gotseff, J. Cale, M. Baggu, D. Narang, and K. Carroll, "Accurate power prediction of spatially distributed pv systems using localized irradiance measurements," in *Proc. IEEE PES General Meeting Conf. Expo.* IEEE, 2014, pp. 1–5.

[27] 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 Trans. Power Syst.*, vol. 33, no. 3, pp. 3255–3264, 2017.

[28] C. Dinesh, S. Welikala, Y. Liyanage, M. P. B. Ekanayake, R. I. Godaliyadda, and J. Ekanayake, "Non-intrusive load monitoring under residential solar power influx," *Appl. Energy*, vol. 205, pp. 1068 – 1080, 2017.

[29] E. C. Kara *et al.*, "Disaggregating solar generation from feeder-level measurements," *Sustain. Energy Grids.*, vol. 13, pp. 112–121, Nov. 2018.

[30] F. Wang, K. Li, X. Wang, L. Jiang, J. Ren, Z. Mi, M. Shafie-khah, and P. Catalão, João, "A distributed pv system capacity estimation approach based on support vector machine with customer net load curve features," *Energies*, vol. 11, no. 7, p. 1750, 2018.

[31] F. Sossan, L. Nespoli, V. Medici, and M. Paolone, "Unsupervised disaggregation of photovoltaic production from composite power flow measurements of heterogeneous prosumers," *IEEE Trans. Ind. Informat.*, vol. 14, no. 9, pp. 3904–3913, 2018.

[32] H. Shaker, H. Zareipour, and D. Wood, "Estimating power generation of invisible solar sites using publicly available data," *IEEE Trans. Smart Grid*, vol. 7, no. 5, pp. 2456–2465, 2016.

[33] 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 Trans. Power Syst.*, pp. 1–1, 2020.

[34] 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.

[35] R. Saeedi, S. K. Sadanandan, A. Srivastava, K. Davies, and A. Gebremedhin, "An adaptive machine learning framework for behind-the-meter load/pv disaggregation," *IEEE Transactions on Industrial Informatics*, 2021.

[36] G. S. Ledva, L. Balzano, and J. L. Mathieu, "Real-time energy disaggregation of a distribution feeder's demand using online learning," *IEEE Trans. Power Syst.*, vol. 33, no. 5, pp. 4730–4740, Jan. 2018.

[37] S. Wang, R. Li, A. Evans, and F. Li, "Regional nonintrusive load monitoring for low voltage substations and distributed energy resources," *Applied Energy*, vol. 260, p. 114225, 2020.

[38] Z. Fang, D. Zhao, C. Chen, Y. Li, and Y. Tian, "Nonintrusive appliance identification with appliance-specific networks," *IEEE Trans. Ind. Appl.*, vol. 56, no. 4, pp. 3443–3452, 2020.

[39] 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.

[40] M. Zhong, N. Goddard, and C. Sutton, "Signal aggregate constraints in additive factorial HMMs, with application to energy disaggregation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3590–3598.

[41] T. Ji, L. Liu, T. Wang, W. Lin, M. Li, and Q. Wu, "Non-intrusive load monitoring using additive factorial approximate maximum a posteriori based on iterative fuzzy *c*-means," *IEEE Trans. Smart Grid*, vol. 10, no. 6, pp. 6667–6677, 2019.

[42] E. Elhamifar and S. Sastry, "Energy disaggregation via learning 'powerlets' and sparse coding," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 629–635.

[43] S. Singh and A. Majumdar, "Deep sparse coding for non–intrusive load monitoring," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 4669–4678, Feb. 2017.

[44] F. M. Wittmann, J. C. López, and M. J. Rider, "Nonintrusive load monitoring algorithm using mixed-integer linear programming," *IEEE Transactions on Consumer Electronics*, vol. 64, no. 2, pp. 180–187, 2018.

[45] A. Rahimpour, H. Qi, D. Fugate, and T. Kuruganti, "Non-intrusive energy disaggregation using non-negative matrix factorization with sum-to-k constraint," *IEEE Trans. Power Syst.*, vol. 32, no. 6, pp. 4430–4441, April 2017.

[46] A. Miyasawa, Y. Fujimoto, and Y. Hayashi, "Energy disaggregation based on smart metering data via semi-binary nonnegative matrix factorization," *Energy and Buildings*, vol. 183, pp. 547–558, 2019.

[47] X. He, L. Chu, R. C. Qiu, Q. Ai, Z. Ling, and J. Zhang, "Invisible units detection and estimation based on random matrix theory," *IEEE Trans. Power Syst.*, vol. 35, no. 3, pp. 1846–1855, 2019.

[48] X. Zhang and S. Grijalva, "A data-driven approach for detection and estimation of residential pv installations," *IEEE Trans. Smart Grid*, vol. 7, no. 5, pp. 2477–2485, 2016.

[49] S. Hosur and D. Duan, "Subspace-driven output-only based change-point detection in power systems," *IEEE Trans. Power Syst.*, vol. 34, no. 2, pp. 1068–1076, 2018.

[50] I. Yildirim, "Bayesian inference: Gibbs sampling," *Technical Note, University of Rochester*, 2012.

[51] J. Paisley, D. M. Blei, and M. I. Jordan, "Variational bayesian inference with stochastic search," in *Proc. 29th Int. Conf. Mach. Learn.*, 2012, pp. 1363–1370.

[52] D. T. Nguyen, M. Negnevitsky, and M. De Groot, “Pool-based demand response exchange—concept and modeling,” *IEEE Trans. Power Syst.*, vol. 26, no. 3, pp. 1677–1685, 2010.

[53] C. Draxl, A. Clifton, B.-M. Hodge, and J. McCaa, “The wind integration national dataset (wind) toolkit,” *Appl. Energy*, vol. 151, pp. 355–366, July 2015.

Ming Yi (S’17) received the B.E. degree in automation from Harbin Engineering University, Harbin, China, in 2016, and the M.S. degrees in control science and engineering from Harbin Institute of Technology, Harbin, China, in 2018, respectively.

He is currently a Ph.D. student in Rensselaer Polytechnic Institute, Troy, NY, USA. His research interests include signal processing, machine learning, power systems monitoring, and high dimensional data analysis.

Meng Wang (M’12) received B.S. and M.S. degrees from Tsinghua University, China, in 2005 and 2007, respectively. She received the Ph.D. degree from Cornell University, Ithaca, NY, USA, in 2012.

She is an Associate Professor in the department of Electrical, Computer, and Systems Engineering at Rensselaer Polytechnic Institute, Troy, NY, USA. Her research interests include high-dimensional data analytics, machine learning, power systems monitoring, and synchrophasor technologies.