An Efficient and Accurate Nonintrusive Load Monitoring Scheme for Power Consumption

Jialing He, Zijian Zhang, *Member, IEEE*, Liehuang Zhu, *Member, IEEE*, Zhesi Zhu, Jiamou Liu, and Keke Gai

***Abstract*—Nonintrusive load monitoring (NILM) has attracted tremendous attention owing to its cost efficiency in electricity and sustainable development. NILM aims at acquiring individual appliance power consumption rates using an aggregated power smart meter reading. Each individual appliance's power consumption enables users to monitor their electricity usage habits for rational saving strategies. This is also a valuable tool for detecting failure in appliances. However, the major barriers facing NILM schemes are issues of accurately capturing the features of each appliance and decreasing the computing time. Motivated by these challenges, we propose a new, efficient, and accurate NILM scheme, consisting of a learning step and a decomposing step. In the learning step, we propose the fast search-and-find of density peaks (FSFDPs) clustering algorithm aimed at capturing the features of the power consumption patterns of appliances. In the decomposing step, we propose a genetic algorithm (GA)-based matching algorithm to estimate the power consumption of each individual appliance using the aggregated power reading. Using elitist and catastrophic strategies, this step reduces the searching space to achieve considerable efficiency. Experimental results using the reference energy disaggregation dataset (REDD) indicate that our proposed scheme promotes accuracy by 10% and reduces the decomposing time by half.**

***Index Terms*—Energy conservation, energy disaggregation, feature learning, nonintrusive load monitoring (NILM), supervised.**

I. Introduction

ENERGY saving is obtained from the electricity consumption data of each individual appliance. These data help users to estimate their electricity usage patterns and adjust their strategies to save energy. Previous studies show that this strategy can help users to save household electricity consumption from 5% to 20% [1]–[7]. Furthermore, the power consumption information of each appliance is valuable for detecting the failure of each appliance [8]–[10]. A great deal of research has focused on this domain [11]–[17]. Even though smart meters are deployed to record household power consumption, only aggregated electricity consumption is recorded.

Therefore, how to recover each appliance's power consumption from an aggregated power reading recorded by a smart meter becomes a concerning problem, which is also the problem to be tackled in this paper. Traditionally, intrusive load monitoring (ILM) sensors were connected to each appliance to collect the power consumption; however, an ILM-based power system with several appliances requires massive sensors, which incurs a prohibitive deployment cost. Besides the ILM approach, the nonintrusive load monitoring (NILM) scheme draws inferences from the power consumption data of each appliance from a single source (smart meter) instead of distributed sensors. NILM or energy disaggregation can decompose aggregated power consumption data into individual appliances' power consumption through the appliance power consumption patterns. Hart [18] proposed the NILM framework, estimating individual load characteristics by analyzing the current and voltage patterns of the aggregated load.

Inspired by the initial framework, two kind of NILM-based schemes have been proposed: 1) supervised and 2) unsupervised. The main differences between the two are the learning features of appliances. The supervised schemes [19]–[24] use training datasets of each individual appliance to obtain feature matrices and to construct a feature dictionary. An appliance can find a linear combination of features in the dictionary. Instead of using individual appliance training data, unsupervised algorithms [25]–[27] learn the features through the dataset of the aggregated power consumption. Yet, manual labels of the appliances are required after the learning stage. This may degrade the decomposing accuracy and efficiency. In this paper, we consider only supervised NILM approaches.

Several challenges were encountered in designing NILM schemes.

1) In the learning stages of the complicated time-series training data of appliances, inaccurate features may be captured, which degrades accurate individual appliance estimation from the aggregated power consumption data.
2) To capture all features, a training dataset with a long span of time is required, which may significantly degrade the efficiency of the scheme.

Manuscript received March 18, 2019; revised May 26, 2019; accepted June 27, 2019. Date of publication July 4, 2019; date of current version October 8, 2019. This work was supported in part by the China National Key Research and Development Program under Grant 2016YFB0800301, in part by the National Natural Science Foundation of China under Grant 61772070, and in part by the Graduate Technological Innovation Project of Beijing Institute of Technology under Grant 2019CX10014. *(Corresponding authors: Zijian Zhang; Liehuang Zhu.)*

J. He, L. Zhu, Z. Zhu, and K. Gai are with the Beijing Engineering Research Center of Massive Language Information Processing and Cloud Computing Application, School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China (e-mail: liehuangz@bit.edu.cn).

Z. Zhang is with the Beijing Engineering Research Center of Massive Language Information Processing and Cloud Computing Application, School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China, and also with the School of Computer Science, University of Auckland, Auckland 1142, New Zealand (e-mail: zhangzijian@bit.edu.cn).

J. Liu is with the School of Computer Science, University of Auckland, Auckland 1142, New Zealand.

Digital Object Identifier 10.1109/JIOT.2019.2926815

3) The time of matching algorithms of existing NILM schemes is, moreover, relatively long.

We develop some novel insights to address these challenges. First, we adopt sliding windows to preprocess training data and to capture features of contextual information from the sequential training. Second, we use a fast search-and-find of density peaks (FSFDPs) [28] clustering algorithm to learn the features. Using the FSFDP algorithm, the features are clustering centers with high local density, and large distance. The number of clustering centers can be raised intuitively; outliers can be spotted automatically and excluded to reduce the time to learn the features. Third, a new matching algorithm based on the genetic algorithm (GA) [29] is proposed to decompose the aggregated power consumption. Optimization strategies, such as the elitist strategy and the catastrophic strategy can narrow the search space and guarantee a small decomposing time.

The contributions of this paper are summarized as follows.

1) We propose a new learning algorithm based on the FSFDP [28] clustering algorithm to train the dataset of each individual appliance and obtain the appliance power consumption patterns efficiently. In addition, sliding windows are used to process the input datasets to capture more accurate features and to guarantee a higher accuracy for decomposing the aggregated power consumption.
2) We further propose a new matching algorithm based on GA to decompose the aggregated power consumption into its component appliances' power consumptions. The new matching algorithm provides lower decomposing time than the existing schemes [20], [22], [23].
3) We perform experiments based on the reference energy disaggregation dataset (REDD) [30]. The experimental results show that the proposed scheme achieves promising results (the time is reduced by half for disaggregating, and the accuracy is increased by 10%, which is twice the promotion of the efficient sparse coding-based data-mining ESCD scheme [23]).

The rest of this paper is organized as follows. Section II discusses the preliminaries. Section III illustrates the details of our proposed scheme. The performance of our scheme's efficiency and the accuracy of the REDD dataset are evaluated in Section IV. Section V provides the discussion. Section VI reviews related work. Section VII concludes this paper.

II. Preliminaries

Rodriguez and Laio [28] proposed the FSFDP clustering algorithm in 2014, indicating that the core of FSFDP was in the clustering centers. The authors identified the following features for a data point of the clustering center.

1) Its local density ρ is large, i.e., it is surrounded by points with smaller local density.
2) Its distance δ from points with larger density is large.

It assumes $S = \{X_i\}_{i=1}^{N}$ represents the clustered data, where $I_S = \{1, 2, \ldots, N\}$ denotes the corresponding index set. $d_{ij} = \text{dist}(X_i, X_j)$ denotes the distance from X_i to X_j. The local density ρ_i, and the distance δ_i of point X_i are computed

$$\rho_i = \sum_{j \in I_S \setminus \{i\}} \chi(d_{ij} - d_c) \qquad \chi(x) = \begin{cases} 1, & x < 0 \\ 0, & x \geq 0 \end{cases} \tag{1}$$

d_c is a cutoff distance pointed out beforehand. Indeed, ρ_i is the number of data points that are closer than d_c to point X_i

$$\delta_i = \begin{cases} \min\limits_{j \in I_S^i} \{d_{ij}\}, & I_S^i \neq \varnothing \\ \max\limits_{j \in I_S} \{d_{ij}\}, & I_S^i = \varnothing. \end{cases} \tag{2}$$

Equation (2) shows a method of computing δ_i, where $I_S^i = \{k \in I_S : \rho_k > \rho_i\}$.

After computing ρ and δ for each data point, the points with larger ρ and δ are chosen clustering centers. The number of clustering centers is manually determined according to specific requirements.

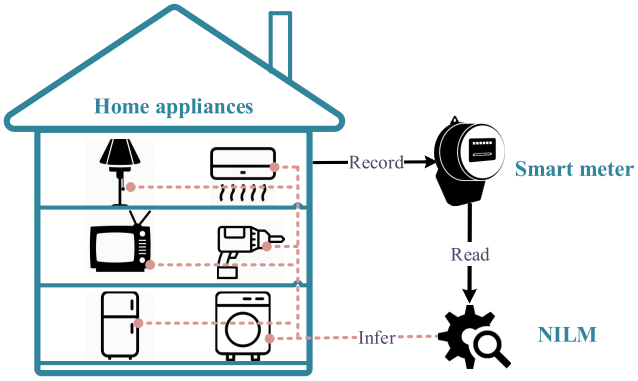


Fig. 1. System model.

III. Our Proposed Scheme

In this section, we describe the details of our scheme (referred to as EAPA) aimed at capturing each individual appliance's real-time power consumption from an aggregated power reading.

A. Model and Problem

The system model (see Fig. 1) consists of three entities: 1) a house with various appliances; 2) a smart meter; and 3) an NILM algorithm. We only consider the power consumption of home appliances. The smart meter records the total power consumption of the house comprising various appliances. The NILM or energy disaggregation algorithm recovers the power consumption of each individual appliance through the aggregated power reading from the smart meter.

We assume that there are N appliances in a house, where $y(t)$ denotes the total power reading of the house (i.e., that contains all N appliances) at time t; $x_i(t)$ represents the power reading of the ith appliance. These are denoted in the following equation:

$$y(t) = \sum_{i=1}^{N} x_i(t). \tag{3}$$

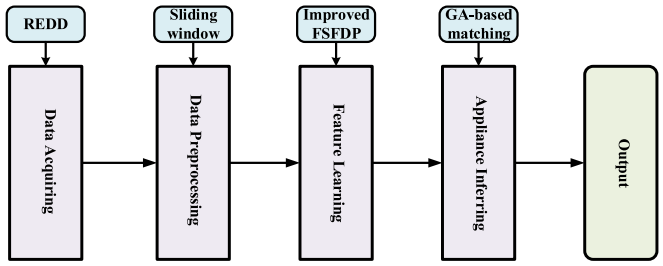


Fig. 2. Flow chart of EAPA.

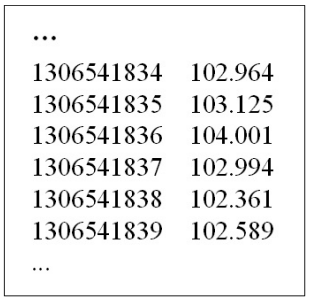


Fig. 3. Original power reading in the REDD dataset.

The problem is how to infer $\{x_i(t)\}_{t=1}^{T}$ where $i \in \{1, 2, \ldots, N\}$ gives the aggregated power reading $\{y(t)\}_{i=1}^{T}$.

Fig. 2 shows that EAPA mainly consists of four steps.

1) *Data Acquiring:* The appliance power consumption data are acquired from the dataset REDD [30], a famous public dataset downloaded from http://redd.csail.mit.edu.
2) *Data Preprocessing:* In this step, we use sliding windows to perform preprocessing for the raw power consumption readings captured from the REDD dataset.
3) *Feature Learning:* We use an improved FSFDP clustering algorithm to learn a feature dictionary that consists of the features of appliances contained in the REDD dataset.
4) *Appliance Inferring:* We propose a GA-based matching algorithm to infer individual appliances from aggregated power consumptions.

B. Data Acquiring

The REDD dataset comprises the power consumption data of real homes and appliances in each house. The dataset consists of high-frequency current/voltage waveform data and low-frequency power data, i.e., 1 Hz power reading. In this paper, we choose the 1 Hz power readings, such as PED [20] and ESCD [23]. The data file contains UTC timestamps (as integers) and power readings as indicated in Fig. 3.

We design a data reading program to get the second column of all data files (the files containing the total reading of the house and each individual appliance' reading), namely, power readings, in order. The total power reading is: $Y = (y(1), y(2), \ldots, y(T))$, where T is time length of power measurements. The individual appliance's power reading is: $X_i = (x_i(1), x_i(2), \ldots, x_i(T))$, where $i \in \{1, 2, \ldots, N\}$.

C. Data Preprocessing

Appliance power consumption data are related to time series. This shows that an appliance power measurement from a high to low value identifies whether the appliance is on or off is a series of sequential data. We address all the data using sliding windows to capture the features better and to obtain each individual appliance's power signal more precisely.

Specifically, the power reading captured in the prior step is split into several s-dimensional vectors, where s represents the length of a sliding window. If the time of training data is T, then, $s \ll T$. After the sliding window-based process, the aggregated power signal and the kth ($k \in \{1, 2, \ldots, N\}$) appliance's power signal can be represented by data points: $q_i = \{y(i), y(i+1), \ldots, y(i+s-1)\}$ and $p_j^k = \{x^k(j), x^k(j+1), \ldots, x^k(j+s-1)\}$, where $i, j \in \{1, 2, \ldots, T-s+1\}$.

D. Feature Learning

In this paper, we utilize an FSFDP-based clustering algorithm to learn the feature matrix of each appliance contained in the training data set similar to [23]. The feature matrix of the ith ($i \in \{1, 2, \ldots, N\}$) appliance is composed of M_i vectors. Each vector corresponds to a clustering center chosen by the clustering algorithm, i.e., a sliding window-based data point $p_k = \{x(k), x(k+1), \ldots, x(k+s-1)\}$ where s is the size of a sliding window and $x(k)$ represents the kth power reading of this appliance. The value of M for different appliances may not the same, i.e., $M_i \neq M_j$, which stands for the number of clustering centers calculated using improved FSFDP clustering algorithm.

Next, we illustrate how to extract the feature matrix for one appliance. All the data points ($p_k = \{x(k), x(k+1), \ldots, x(k+s-1)\}$, where $k \in \{1, 2, \ldots, T-s+1\}$) are checked and unique ones are chosen to form a new data set which is denoted as P_{uni}. The number of the data points in P_{uni} is R. After obtaining the dataset P_{uni}, we calculate a distance dataset $D = \{d_{ij} | i, j \in 1, 2, \ldots, R\}$. This depicts that the distance between the two data points belonging to P_{uni}. Specifically, the distance from p_i to p_j is computed as $d_{ij} = \|P_i - P_j\|_2$, where $d_{ij} = d_{ji}$.

After obtaining P_{uni} and $D = \{d_{ij} | i, j \in 1, 2, \ldots, R\}$, the clustering algorithm is utilized. The core idea of the basic FSFDP algorithm is to determine the clustering centers through two parameters ρ_i and δ_i. The basic FSFDP clustering algorithm selects the data points with larger ρ_i and δ_i as the clustering centers. However, the number of the clustering centers cannot be automatically determined.

From the FSFDP clustering algorithm developed by Wang *et al.* [23], the number of clustering centers was set just through their experience (they set 60 as the number of clustering centers for all the training datasets). Such a manual process is inefficient and lacks objectivity. In their subsequent work, a semiautomated clustering algorithm was proposed. They performed clustering operations twice. In the first round, they chose m_i data points with large ρ_j and $\delta_j (j \in \{1, 2, \ldots, R\})$ as the clustering centers for the ith appliance, where $i \in \{1, 2, \ldots, N\}$, m_i represents the number of the states of the ith appliance's consumption power signal (Algorithm 1 shows the process of computing m_i). The use of m_i has made the first round of clustering automated. But

Algorithm 1: Function_calm():compute m_i [23]

Input: Appliance training data set D_{ta}, $\mu 1$ and $\mu 2$.
Output: m_i
1. Set $m = 1$, $a = [\quad]$, $a[1] = D_{ta}^{1}$.
2. **for** $i = 2, \ldots, size(D_{ta})$ **do**
3. **if** $(D_{ta}^{i} < D_{ta}^{(i-1)} \cdot \mu 1 \parallel D_{ta}^{i} > D_{ta}^{(i-1)} \cdot \mu 2)$
4. **for** $j = 1, 2, \ldots, m$ **do**
5. **if** $(D_{ta}^{i} > a[j] \cdot \mu 1 \parallel D_{ta}^{i} < a[j] \cdot \mu 2)$
6. D_{ta}^{i} is added into $j - th$ class, and the average value of this class is computed.
7. **else**
8. if $j < m$ $m = m + 1$. D_{ta}^{i} is a new class a[m]=D_{ta}^{i}.
9. **end if**
10. **end for**
11. **end if**
12. **end for**

they did not demonstrate their reason for doing like this. In the second round, each class was further divided into 20 clusters, where 20 was set based on their experience. Although this semiautomated clustering algorithm has enhanced the efficiency of the basic FSFDP algorithm, it still depended on human's experience, which was inefficient and unreasonable. In this paper, an adaptive learning algorithm based on FSFDP clustering is proposed to learn the feature matrix in a more efficient and rigorous way.

In our new scheme, the number of clustering centers (i.e., M_i) is determined automatically based on the states of each appliance and the size of the sliding windows. Exploiting these parameters to learn the feature matrix is more reasonable and can increase the decomposing accuracy in the next stage.

Using our improved clustering algorithm, we cluster all the data points twice. We compute ρ_i and δ_i for the data points of the involved appliance. For a data point p_i, ρ_i is the local density, and δ_i is the distance from p_i to other points of higher density. ρ_i and δ_i are computed as shown in (4) and (5), respectively

$$\rho_i = \sum_{j \in I_S \setminus \{i\}} e^{-\left(\frac{d_{ij}}{d_c}\right)^2} \tag{4}$$

$$\delta_i = \begin{cases} \min\limits_{j \in I_S^i} \{d_{ij}\}, & I_S^i \neq \varnothing \\ \max\limits_{j \in I_S} \{d_{ij}\}, & I_S^i = \varnothing. \end{cases} \tag{5}$$

In (4), d_c is the cutoff distance set beforehand. In our scheme, d_c is calculated as follows. We arrange all the distances d_{ij} in a ascending order to obtain a sequence; d_c is then set as the value of the middle. In the equation, the more data points whose distances from p_i are less than d_c, the greater the value of ρ_i. Equation (1) shows the original way of computing ρ_i proposed by Rodriguez and Laio [28]. Compared with the value of ρ_i obtained from (1), which is discrete, ρ_i captured from (4) proposed in this paper is continuous. This shows that ρ_i captured from (4) is less likely to conflict (conflict means different points may have the same value of ρ). This helps in determining the clustering centers and promoting the accuracy of our scheme. For example, 1000 data points are assumed to be clustered, 10 of them with larger ρ are clustering centers. The point with the largest ρ is chosen first; this is repeated until the 10th clustering center. During the choosing process, there is a high probability that two points with the same ρ are computed by (1). This may impede the selection for the right clustering centers. Equation (4) follows the sharing process of the original FSFDP algorithm, indicating that the more the data points closer to X_i than d_c, the larger the ρ_i. In (5), $S = \{p_i\}_{i=1}^{R}$ contains all the clustered data points, $I_S = \{1, 2, \ldots, R\}$ is set of relative indices. There exists the equation that $I_S^i = \{k \in I_S : \rho_k > \rho_i\}$. So, δ_i in (5) means the minimum distance between p_i and the other data points with higher ρ_i.

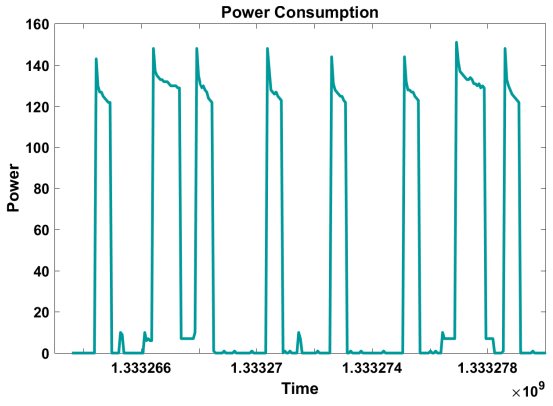


Fig. 4. Power consumption signal of a refrigerator.

In the first clustering process, we choose m_i data points with large ρ_j and $\delta_j (j \in \{1, 2, \ldots, R\})$ as the clustering centers for the ith appliance, where $i \in \{1, 2, \ldots, N\}$. Similar to Wang *et al.*'s work [23], m_i represents the number of the states of the ith appliance's consumption power signal. The reason to use m_i is that by observing the power consumption waveform (see Fig. 4) of appliances, it can be seen that the similar states of the power consumption can produce the similar fluctuations, where m_i denotes the number of all the states. All the m_i unique states can represent an appliance's power consumption patterns. Specifically, we choose the data points with the largest value of ρ_j and δ_j as the first clustering center. Then the values of ρ_j and δ_j are reduced to select clustering centers. The values are reduced by 1% each time until the number of clustering centers is equal or larger to m_i. After the first clustering process, all the data points of the ith appliance are divided into m_i classes.

In the second round of our new scheme, each class is divided into s clusters. Here, s is the size of a sliding window. We perform this process because all the data are processed in units of sliding windows, which implies that the classes of an appliance's data (i.e., M_i) are highly probabilistically correlated to the size of a sliding window. After the two clustering processes, all the data points of the ith appliance is clustered into $M_i = m_i \times s$ classes. Algorithm 2 demonstrates the entire learning process.

Algorithm 2: Learning the Feature Matrix

Input: Appliance training dataset D_{ta}, the size of a sliding window s,
Output: Appliance feature matrix F.
1. $m =$ Funtion_calm(0.5, 1.5);
2. Obtain P_{uni} and R of D_{ta}.
3. Obtain the distance matrix D.
4. **for**$i, j = 1, 2, \ldots, R$**do**
5. $d_{ij} = \|p_i - p_j\|_2$
6. **end for**
7. First clustering $FSFDP(D, m)$.
8. **for** $i = 1, 2, \ldots, m$ **do**
9. $M_i = m \times s$
10. $FSFDP(D_i, M_i)$
11. Clustering result is collected to F.
12. **end for**

7,7,7,7,7,7,150,138,137,135,134,133,132,132,132,131,131,130,130,129
133,132,132,132,131,130,130,130,130,129,129,128,128,128,7,7,6,7,0,0
0,147,134,129,128,126,125,124,123,122,122,0,1,0,0,0,0,0,0,0
7,7,7,7,7,7,7,150,139,136,134,133,134,133,132,131,131,130,130,129
7,7,7,7,7,150,138,137,135,134,133,132,132,132,131,131,130,130,129,129
7,7,7,7,150,138,137,135,134,133,132,132,132,131,131,130,130,129,129,129
7,7,7,150,138,137,135,134,133,132,132,132,131,131,130,130,129,129,129,128
7,7,7,7,7,7,7,7,150,138,136,135,134,133,133,133,131,131,130,130
7,7,150,138,137,135,134,133,132,132,132,131,131,130,130,129,129,129,128,128
7,152,141,136,135,134,133,132,132,132,130,130,131,130,130,129,129,129,128,128

Fig. 5. Part of the feature matrix of a refrigerator computed by our improved FSFDP clustering algorithm (all the numbers in this figure are the results of rounding).

Fig. 5 shows part of the feature matrix of a refrigerator in the training dataset. Where one row represents one feature (in this example, the feature is a 20-D vector), each number denotes the power reading of each measuring time.

E. Appliance Inferring

Appliance inferring is actually to decompose the aggregated power reading according to the feature matrices obtained from the prior step.

After acquiring the feature matrix F_i of each individual appliance in the former step, we can approximately denote $T_s x(t) \approx F_i \times a_i(t)$. Then $x(t)$ is recovered from F_i and $a_i(t)$. This is the goal of energy disaggregation. The goal can be further described equivalently as computing $a_i(t)$ for each appliance given the feature matrix F_i and the constraint that $T_s x_i(t) \approx F_i \times a_i(t)$ and $T_s y(t) = \sum_{i=1}^{N} T_s x_i(t)$. On the basis of our assumption that $a_i(t)$ is one nonezero element of 1 and the other elements are all 0, therefore, $F_i \times a_i(t)$ represents one row vector in the matrix of F_i.

From another point of view, the goal of this step is actually a problem to seek a global optimal solution. Suppose, there are N appliances, and the number of the rows of F_i is M_i, hence, the complexity to solve the problem is $O(\prod_{i=1}^{N} M_i)$. Without optimization, this complexity is unacceptable. Elhamifar and Sastry [20] proposed an optimization algorithm that decomposed the aggregated power consumption signal. This was achieved by searching for an appropriate representation of the individual appliances. Each disaggregation process for a sliding windows power reading was estimated as 12 s. Using pruning operations, Wang *et al.* [23] used an MMPM matching algorithm, to optimize the disaggregation process. In their scheme, the disaggregation process was approximately 10.7 s shorter compared with that achieved by [20]. In this paper, we propose a new matching algorithm based on the GA algorithm to perform the decomposing process.

The matching algorithm is mainly composed of five parts that are chromosomes encoding and the initial population producing; the fitness computing for each chromosome; chromosomes selecting from the current population; chromosomes crossing; and chromosomes mutation. These critical procedures will now be discussed in detail.

To find a solution of N vectors, we decompose the aggregated power consumption data. Each vector from a feature matrix represents an appliance. For each vector, the ordinal number of the vector in the feature matrix, namely, the number of the row uniquely identifies the vector. Therefore, we connect the ordinal number of each vector as the solution.

We choose the gray code to encode each solution that is regarded as one chromosome and one gene of the chromosome represents one bit. The number of the total genes in a chromosome is *num_cc*; let $num\_c_i, i \in \{1, 2, \ldots, N\}$ represent the number of the genes of each vector. Then $num\_c_i, i \in \{1, 2, \ldots, N\}$ is determined by the size of M_i (the number of the rows for the ith feature matrix), which can be denoted as $\lceil M_i/2 \rceil$. We generate some chromosomes to form an initial population randomly, the number of the population is *num_pop*. To obtain the optimal solution rapidly, we verify whether the random chromosome is a feasible solution. If it is not, it will be abandoned because the probability of an unfeasible solution evolving into a feasible solution is negligible. Here, a feasible solution is the sum of all vectors equal to or smaller than the aggregated power Y, which is the power consumption data to be decomposed.

The fitness function in this paper is the accuracy function demonstrated in (6). After the initial population is determined, the algorithm the algorithm enters an iterative process and generates populations. The number of the iterations is *num_ite*. When the process iterate *num_ite* times, the matching process ceases and outputs the current optimal solution. For the current population, the fitness function is used to calculate the fitness for each chromosome in the population

$$f(x_i) = 1 - \frac{\sum_{t\in\varphi} \sum_{i=1}^{N} \left\| T_w(y(t)) - \tilde{T_w}(x_i(t)) \right\|_1}{2 \sum_{t\in\varphi} \|T_w(y(t))\|_1}. \quad (6)$$

For example, if one of the chromosomes in the current generation is the chromosome shown in Fig. 6, the fitness of which

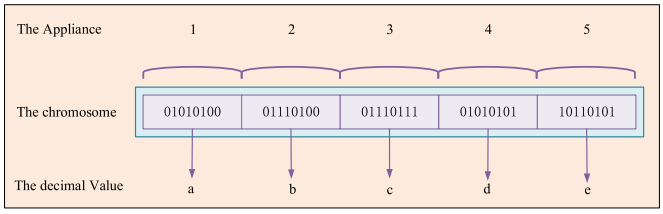


Fig. 6. It consists of five appliances, each comprises 8 bit genes. Each part corresponds the line number of the appliance's feature matrix.

is calculated as follows:

$$f = 1 - \frac{\sum_{t}^{t+s-1}\sum_{i=1}^{N}(y(t) - x_i(t))}{2\sum_{t}^{t+s-1} y(t)} = 1 - \frac{\sum_{t}^{t+14}\left(y(t) - F_1^a(t) - F_2^b(t) - F_3^c(t) - F_4^d(t) - F_5^e(t)\right)}{2\sum_{t}^{t+14} y(t)}$$

where $s = 15$ denotes the size of the sliding window, F_1^a represents the ath row of feature matrix F_1, F_2^b represents the bth row of feature matrix F_2, F_3^c represents the cth row of feature matrix F_3, F_4^d represents the dth row of feature matrix F_4, and F_5^e represents the eth row of feature matrix F_5.

We generate the next population according to the fitness of each chromosome. The process consists of chromosome selecting, chromosomes crossing, and chromosome mutation. The roulette algorithm is the selected method. We use the fitness of each chromosome to compute the probability. We calculate two probabilities for each chromosome. The first is the probability that chromosomes are inherited to the next generation, which is denoted as $p(x_i) = f(x_i)/(\sum_{j=1}^{num\_pop} f(x_j))$. Second is the cumulative probability that is denoted as $q_i = \sum_{j=1}^{i} p(x_j)$. We will choose a random number $r \in (0, 1)$, and determine the chromosomes that inherited in the next generation according to r and q_i. If q_i is exactly equal or greater than r, then the x_i chromosome is selected. We execute the selected process $num\_pop/2$ times and select two chromosomes each time.

We cross the two chromosomes using the specific crossover method, which is a single-point crossover. We note every gap between two genes in a chromosome as a point. With $num\_cc - 1$ points for each chromosome, we randomly select one point and exchange the genes in the two chromosomes before and after the point to get two new chromosomes. At the same time, we set the probability of crossing two chromosomes as p_cro.

For the two new generated chromosomes, mutation operations are executed. The mutation method in this paper is the single-point mutation, and each chromosome mutates one gene from 1 to 0 or 0 to 1. The probability of mutation is p_mut.

Optimization strategies, such as the elitist strategy and the catastrophic strategy promote the efficiency of the matching algorithm. The elitist strategy helps to copy the best solution for the current generation to the next generation. It can avoid losing the best solution in the end. The catastrophic strategy helps to cutoff generations if the best solution of the current generation is the same within 20 generations. The catastrophic strategy helps the matching algorithm to avoid local optimization. Algorithm 3 provides the matching algorithm for decomposing the aggregated power consumption data.

Algorithm 3: Decompose the Aggregated Data

Input: Feature matrix $F_i, i \in \{1, 2, \ldots, N\}$, total power signal Y.
Output: The best optimal solution *result*.
1.Set $num\_pop = 400$, $num\_ite = 600$, $p\_cro = 0.9$, $p\_mut = 0.05$, $flag = 1$
2.**for** $i = 1, \ldots,$ *num_pop* **do**
Randomly generate chromosomes $ch[i]$ according to vectors in F_i and Y and form the current population *POP_current*.
3.**end for**
4.Set $i = 0$
5.**while**$\{i <$ *num_ite*$\}$
6. **for** $j = 1, \ldots,$ *num_pop*
6. Compute fitness for *POP_current*.
7. *fit*(*ch*[*j*])
8. **if** *result.fit* $<$ *ch*[*j*].*fit*
9. *result* = *ch*[*j*]
10. **end if**
11. Choose two chromosomes according to roulette algorithm.
12. **if** *random*(0, 1) $<$ *p_cro*
13. Cross the two chromosomes and derive two new chromosomes.
14. **end if**
15. **if** *random*(0, 1) $<$ *p_mut*
16. Mutate the chromosomes.
17. **end if**
18. Add the new chromosomes to the next population-*POP_next*.
19. **end for**
20. **if** *POP_next.result* == *POP_current.result*
21. flag=flag+1
22. **end if**
23. **else** *flag* = 0
24. **if** *flag* == 20
25. Stop the loop and reconstruct the initial population.
26. **end if**
27.**end while**
28.Ouput *result*.

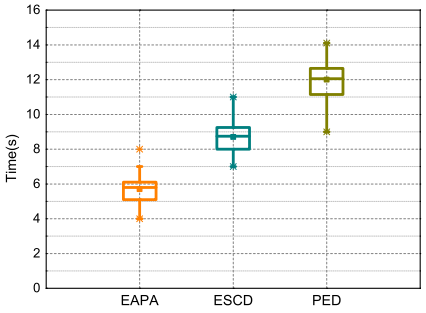
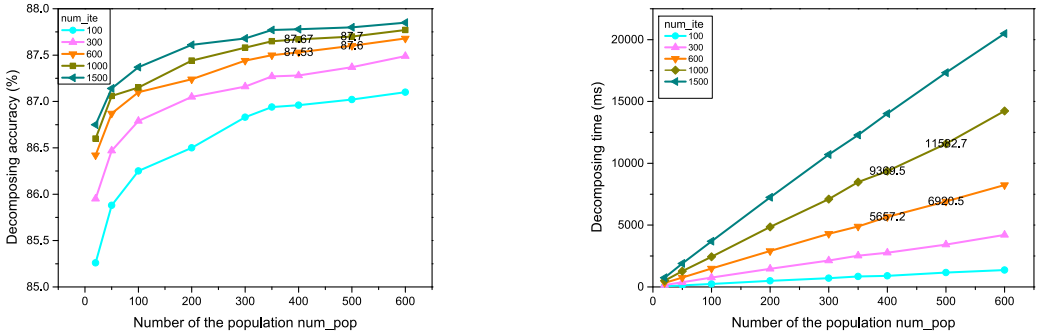


Fig. 7. Decomposing time of each sliding window between different schemes.



(a) (b)

Fig. 8. Performance of a different number of iterations. (a) Decomposing accuracy for different number of iterations. (b) Decomposing time for different number of iterations.

TABLE I
ACCURACY OF ENERGY DISAGGREGATION (%)

	House1	House2	House3	House4	House6	Average
Simple	41.4	39.0	46.7	52.7	33.7	42.7
FHMM	71.5	59.6	59.6	69.0	62.9	64.5
PED	81.6	79.0	61.8	58.5	79.1	72.0
ESCD	84.3	82.7	70.2	71.0	78.9	77.4
EAPA	94.5	91.9	80.3	81.4	89.2	87.5

IV. EVALUATION PERFORMANCES

In this section, we demonstrate the results of experiments preformed on the public dataset REDD [30] which is utilized in [20] and [23] to verify the efficiency and accuracy of our proposed scheme–EAPA.

REDD contains both household aggregated power consumption data and individual appliance data. The power consumption data are from six houses. The power consumption data of the 5th house are excluded because the recording readings of this house contain very few events, which does not help in acquiring appliances' features. REDD comprises both high-frequency and low-frequency data. We exploit the low-frequency data like [20] and [23]. We also use the first-week data to learn the feature matrix and the other data for testing. The size of the sliding windows is s. To capture the transient features of appliances better, we set $s \in [10, 50]$ because the sampling frequency is 1 Hz.

The accuracy is denoted in (6), where $\tilde{T_w}(x_i(t))$ represents the ith appliance's power and $\varphi = \{1, T_w + 1, 2T_w + 1, \ldots\}$.

For the decomposing accuracy, we compare the average accuracy of our scheme with that of other schemes like PED in [20], ESCD in [23], FHMM in [30], and the naive simple mean method. Table I depicts the concrete result, from which we can see that our scheme EAPA can get a promising experiment result that the accuracy can be raised about 10% compared to the ESCD scheme. For the computing time, we evaluate the time of decomposing each sliding window's aggregated power reading between different schemes. Specifically, we repeat the experiment 20 times for each scheme and record each decomposing time. Fig. 7 demonstrates the result: the average time of our proposed scheme (EAPA) is 5.7 s, which is shorter compared with the 8.7 s in ESCD and the 12 s in PED. EAPA slashes the decomposing time especially for numerous sliding windows in practice.

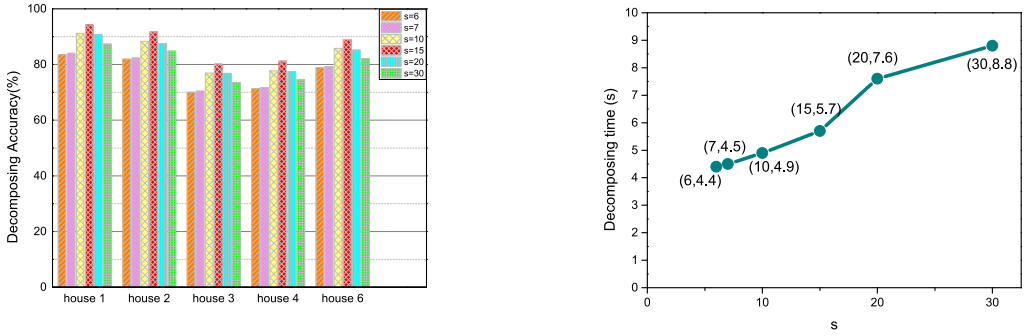
The experimental results show that the size of the population (*num_pop*) and the number of the iterations (*num_ite*) of the decomposing algorithm can affect the accuracy and time of energy disaggregation. Fig. 8 depicts the results. Fig. 8(a) shows that for a fixed *num_ite*, the decomposing accuracy increases with the increase of *num_pop*, and for a fixed *num_pop*, accuracy increases with the increase of *num_ite*. The greater the *num_pop* and *num_ite*, the smaller the rate of the accuracy growth. Fig. 8(b) demonstrates that the decomposing time increases with the increase of *num_pop* and *num_ite*. Regarding the accuracy and efficiency of our scheme, the highest accuracy is achieved within a short time. We set the size of the population to 400 and the number of iterations to 600 for the matching algorithm. At the same time the probability of crossing two chromosomes is 90% and the mutation probability for a gene is 5%. In Section III-D, we have obtained the size of the features in the feature matrix for every appliance is $M_i = m \times s \times \sum_{j=1}^{m} p_j (i \in \{1, 2, \ldots, N\})$. Experimental results show that the size of s can change the accuracy of the disaggregation in the end. Fig. 9 shows the disaggregation accuracy and the time to disaggregate a sliding window's reading for different sizes of s. Fig. 9(a) shows the energy disaggregation accuracy for different sizes of sliding windows. This depicts that the accuracy increases with an increase in s and reaches the highest value when $s = 15$, and then the accuracy decreases with an increase in s. Fig. 9(b) demonstrates the time of decomposing a sliding window of aggregated power signal. It shows that the decomposing time increases with the increase of s. From Fig. 9(a) and (b), it can be seen that when $s = 15$, the accuracy of energy disaggregation is the highest, and the decomposing time is relatively short at the same time. Consequently, we set the size of the sliding window as $s = 15$ in this paper.

Fig. 10 shows the actual power data signal of the aggregated data and the estimated power signal at a disaggregation accuracy of 91.9% in house 2.

V. DISCUSSION

Some other analysis about the GA in our scheme is discussed in this section.

1) *Why GA Is Chosen Other Than ACO or Simulated Annealing?* GA, ACO, and simulated annealing are common approaches for solving optimization



(a) (b)

Fig. 9. Performance for different size of the sliding windows (s). (a) Decomposing accuracy for different size of sliding windows. (b) Decomposing time for different size of sliding windows.

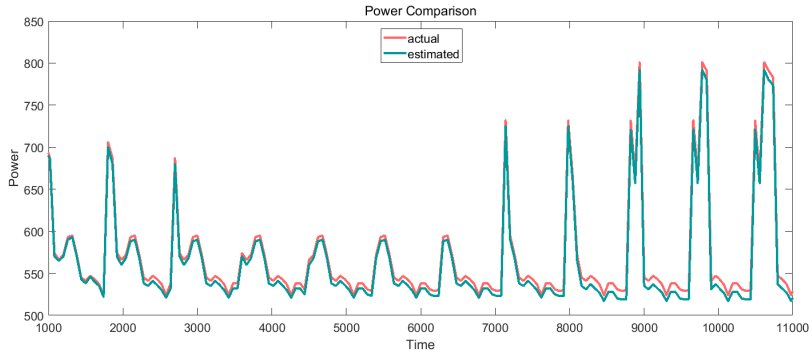


Fig. 10. Estimated power consumption signal and actual power signal in the house 2.

problems: ACO has a relatively slow convergence speed and is easy to fall into local optimal, which is more suitable for searching path on graph; simulated annealing is a random search algorithm, of which the higher the initial temperature T is, the greater the possibility of finding the global optimal solution is, but the computational complexity is also significantly increased; GA is suitable for solving discrete problems and has strong global search ability. Considering the discrete optimization problem of this paper and the accuracy and efficiency requirements, GA is selected.

2) *Gray Code Versus Simple Binary Code:* In our scheme, we choose the gray code to encode the chromosome because of the stability of its coding mode, which can enhance the efficiency and accuracy of GA. The gray code is a reflected binary code where two successive values differ in only one bit. For example, three-bit simple binary codes are: 001 010 011 100 101 110 111, whereas gray codes are: 000 001 011 010 110 111 101 100, showing that a change of simple binary code is not continuous; i.e., two successive values (such as 011 and 100) often differ not only in one bit. When the solution of GA algorithm approaches the optimal solution, the phenotype of the GA algorithm varies considerably and discontinuously after using simple binary coding mutation, which is far away from the optimal solution. Gray codes can be used to avoid this problem very well. Several corresponding studies [31]–[33] certify that gray codes are generally superior to simple binary codes in function optimization when using the GA algorithm.

3) *Elitist and Catastrophic Strategies:* The elitist strategy allows us copying the best solution of the current generation and pass it on to the next generation, which can prevent the optimal solution from being destroyed by crossover and mutation in the evolutionary process. The catastrophic strategy allows us cutting off the generations if the best solution of the current generation has not been changed for a continuous G (in this paper, G is 20) generations, which can avoid entering into local optimum. For these two optimization strategies, we performed four groups of comparative experiments (see Fig. 11), from which we can see that the elitist and catastrophic strategies can markedly promote the accuracy of GA and scarcely increase the decomposing time.

VI. Related Work

NILM or energy disaggregation infers the power consumption of each individual appliance from an aggregated power signal monitored by a smart meter. There are a number of approaches to address energy disaggregation using supervised and unsupervised approaches.

Supervised approaches normally require the training dataset of the individual appliance to capture specific features for each appliance. These features can be utilized to infer different specific appliances from an aggregated power consumption signal. Unsupervised approaches just require a dataset of

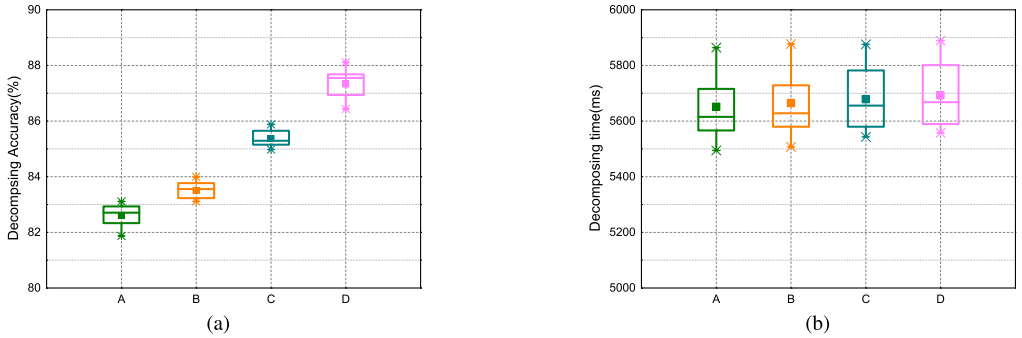


Fig. 11. Performance for utilizing optimization strategies. (a) Simple GA without any optimization strategies. (b) GA with the elitist strategy. (c) GA with the catastrophic strategy. (d) GA with the elitist and catastrophic strategies. All the experiments (20 times) conducted under the condition that $num_{ite} = 600$ (iterations number) and $num_pop = 400$ (population size).

the aggregated power consumption to perform training process, which seems convenient. However, these approaches should label the training result manually, leading to inaccurate decomposing results with difficulty in practical application.

Several existing supervised methods are based on sparse coding [19], [20], [22], [23]. Kolter *et al.* [19] modeled appliances that have a long history of signal data to capture their different times. Using a feature dictionary, disaggregate the aggregated power consumption if possible. The weakness of this approach is its large training dataset requirement that contains an individual appliances signals at all possible times. This large dataset may degrade the efficiency of the scheme. Elhamifar and Sastry [20] proposed a novel sparse coding based on the NILM scheme. A dictionary referred to as "powerlets" was formed in the scheme. The dictionary represents signatures that correspond to the consumption patterns of different appliances automatically extracted using a convex programming scheme. The authors also use an optimization matching algorithm to decompose the aggregated power consumption data. This scheme was free of local issues, and small training data on a single home were enough for the training process. Wang *et al.* [23] improved this scheme in [20] and proposed an efficient sparse coding-based energy disaggregation method. They utilized a novel clustering algorithm-clustering by FSFDPs [28] to train the data set of each individual appliance. However, the manual application of some parameters in the scheme is required; this may cause inefficient and unreasonable results.

VII. Conclusion

We proposed a new, efficient, and accurate supervised energy disaggregation method consisting of a learning stage and a decomposing stage. In the learning stage, we proposed a new learning algorithm based on the FSFDP clustering algorithm to learn the feature matrix of each individual appliance. We use sliding windows to preprocess the sequential training data to capture more accurate features. We also use the number of states that represent the patterns of the appliance and the size of the sliding window to specify the number of cluster centers in the learning algorithm. The numbers of rows in the feature matrix were determined automatically to improve the disaggregation accuracy of energy. In the decomposing stage, we proposed a new matching algorithm based on GA to decompose the aggregated power consumption, which could greatly reduce the searching space thus shortening the decomposing time. Experimental results show that our proposed scheme achieves promising results in both the accuracy and the efficiency in energy disaggregation.

References

[1] A. Faruqui, S. Sergici, and A. Sharif, "The impact of informational feedback on energy consumption—A survey of the experimental evidence," *Energy*, vol. 35, no. 4, pp. 1598–1608, 2010.

[2] A. Barbu, N. Griffiths, and G. Morton, *Achieving Energy Efficiency Through Behavior Change: What Does It Take*, Eur. Environ. Agency, Copenhagen, Denmark, 2013.

[3] S. Darby, "The effectiveness of feedback on energy consumption," *Rev. DEFRA Literature Meter. Billing Direct Displays*, vol. 486, no. 26, 2006.

[4] E.-J. Lee, M.-H. Pae, D.-H. Kim, J.-M. Kim, and J.-Y. Kim, "Literature review of technologies and energy feedback measures impacting on the reduction of building energy consumption," in *Proc. EU-Korea Conf. Sci. Technol. (EKC)*, 2008, pp. 223–228.

[5] M. Li, L. Zhu, and X. Lin, "Efficient and privacy-preserving carpooling using blockchain-assisted vehicular fog computing," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4573–4584, Aug. 2018.

[6] A. Mahone and B. Haley, *Overview of Residential Energy Feedback and Behavior Based Energy Efficiency*, Energy Environ. Econ. Inc., Boston, MA, USA, 2011.

[7] M. E. Berges, E. Goldman, H. S. Matthews, and L. Soibelman, "Enhancing electricity audits in residential buildings with nonintrusive load monitoring," *J. Ind. Ecol.*, vol. 14, no. 5, pp. 844–858, 2010.

[8] Z. Zhang, W. Cao, Z. Qin, L. Zhu, Z. Yu, and K. Ren, "When privacy meets economics: Enabling differentially-private battery-supported meter reporting in smart grid," in *Proc. IEEE/ACM Int. Symp. Qual. Service*, 2017, pp. 1–9.

[9] Z. Zhang, Z. Qin, L. Zhu, J. Weng, and K. Ren, "Cost-friendly differential privacy for smart meters: Exploiting the dual roles of the noise," *IEEE Trans. Smart Grid*, vol. 8, no. 2, pp. 619–626, Mar. 2017.

[10] L. Zhu, M. Li, Z. Zhang, and Z. Qin, "ASAP: An anonymous smart-parking and payment scheme in vehicular networks," *IEEE Trans. Depend. Secure Comput.*, to be published.

[11] A. A. Neghabi, N. J. Navimipour, M. Hosseinzadeh, and A. Rezaee, "Nature-inspired meta-heuristic algorithms for solving the load balancing problem in the software-defined network," *Int. J. Commun. Syst.*, vol. 32, no. 4, 2019, Art. no. e3875.

[12] M. Ashouraei, S. N. Khezr, R. Benlamri, and N. J. Navimipour, "A new SLA-aware load balancing method in the cloud using an improved parallel task scheduling algorithm," in *Proc. IEEE 6th Int. Conf. Future Internet Things Cloud (FiCloud)*, 2018, pp. 71–76.

[13] Z. Ghanbari, N. J. Navimipour, M. Hosseinzadeh, and A. Darwesh, "Resource allocation mechanisms and approaches on the Internet of Things," *Clust. Comput.*, pp. 1–30, Jan. 2019.

[14] M. Hamzei and N. J. Navimipour, "Toward efficient service composition techniques in the Internet of Things," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3774–3787, Oct. 2018.

[15] A. S. Milani and N. J. Navimipour, "Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends," *J. Netw. Comput. Appl.*, vol. 71, pp. 86–98, Aug. 2016.
[16] A. A. Neghabi, N. J. Navimipour, M. Hosseinzadeh, and A. Rezaee, "Load balancing mechanisms in the software defined networks: A systematic and comprehensive review of the literature," *IEEE Access*, vol. 6, pp. 14159–14178, 2018.
[17] B. Pourghebleh and N. J. Navimipour, "Data aggregation mechanisms in the Internet of Things: A systematic review of the literature and recommendations for future research," *J. Netw. Comput. Appl.*, vol. 97, pp. 23–34, Nov. 2017.
[18] G. W. Hart, "Nonintrusive appliance load monitoring," *Proc. IEEE*, vol. 80, no. 12, pp. 1870–1891, Dec. 1992.
[19] J. Z. Kolter, S. Batra, and A. Y. Ng, "Energy disaggregation via discriminative sparse coding," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2010, pp. 1153–1161.
[20] E. Elhamifar and S. Sastry, "Energy disaggregation via learning 'powerlets' and sparse coding," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 629–635.
[21] L. Mauch and B. Yang, "A new approach for supervised power disaggregation by using a deep recurrent LSTM network," in *Proc. Signal Inf. Process.*, 2016, pp. 63–67.
[22] J. Z. Kolter and T. Jaakkola, "Approximate inference in additive factorial HMMs with application to energy disaggregation," in *Proc. Artif. Intell. Stat.*, 2012, pp. 1472–1482.
[23] D. Wang, J. He, M. A. Rahim, Z. Zhang, and L. Zhu, "An efficient sparse coding-based data-mining scheme in smart grid," in *Proc. Int. Conf. Mobile Ad-Hoc Sensor Netw.*, 2017, pp. 133–145.
[24] H. Altrabalsi, J. Liao, L. Stankovic, and V. Stankovic, "A low-complexity energy disaggregation method: Performance and robustness," in *Proc. IEEE Symp. Comput. Intell. Appl. Smart Grid*, 2015, pp. 1–8.
[25] H. Kim, M. Marwah, M. Arlitt, G. Lyon, and J. Han, "Unsupervised disaggregation of low frequency power measurements," in *Proc. Int. Conf. Data Min. SIAM*, 2011, pp. 747–758.
[26] O. Parson, S. Ghosh, M. J. Weal, and A. Rogers, "An unsupervised training method for non-intrusive appliance load monitoring," *Artif. Intell.*, vol. 217, pp. 1–19, Dec. 2014.
[27] H. Gonalves, A. Ocneanu, and M. Bergs, "Unsupervised disaggregation of appliances using aggregated consumption data," in in *Proc. 1st KDD Workshop Data Min. Appl. Sustain. (SustKDD)*, 2011, pp. 1–6.
[28] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.
[29] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence*, vol. 6. Cambridge, MA, USA: MIT Press, 1992, pp. 126–137. 1992.
[30] J. Z. Kolter and M. J. Johnson, "REDD: A public data set for energy disaggregation research," in *Proc. Workshop Data Min. Appl. Sustain. (SIGKDD)*, vol. 25. San Diego, CA, USA, 2011, pp. 1–6.
[31] Y. A. Katsigiannis, P. S. Georgilakis, and E. S. Karapidakis, "Genetic algorithm solution to optimal sizing problem of small autonomous hybrid power systems," in *Proc. Hellenic Conf. Artif. Intell.*, 2010, pp. 327–332.
[32] P. Snaselova and F. Zboril, "Genetic algorithm using theory of chaos," *Procedia Comput. Sci.*, vol. 51, pp. 316–325, Jun. 2015.
[33] R. A. Caruana and J. D. Schaffer, "Representation and hidden bias: Gray vs. binary coding for genetic algorithms," in *Proc. Mach. Learn.*, 1988, pp. 153–161.

Jialing He received the B.Eng. and M.S. degrees from the Beijing Institute of Technology, Beijing, China, in 2016 and 2018, respectively, where she is currently pursuing the Ph.D. degree with the School of Computer Science and Technology.

Her current research interests include cloud security, Internet of Things security, and analysis of entity behavior and preference.

Zijian Zhang (M'14) received the B.Eng., M.S., and Ph.D. degrees from the School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China, in 2006, 2008, and 2012, respectively.

He is an Associate Professor with the School of Computer Science and Technology, Beijing Institute of Technology. He was a Visiting Scholar with the Computer Science and Engineering Department, State University of New York, Buffalo, NY, USA, in 2015. His current research interests include design of authentication and key agreement protocol and analysis of entity behavior and preference.

Liehuang Zhu (M'12) received the B.Eng. and M.S. degrees from Wuhan University, Wuhan, China, in 1998 and 2001, respectively, and the Ph.D. degree from the School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China, in 2004.

He is a Professor with the School of Computer Science and Technology, Beijing Institute of Technology. He is selected into the Program for New Century Excellent Talents in University from the Ministry of Education, Beijing. His current research interests include Internet of Things, cloud computing security, and blockchain.

Zhesi Zhu received the B.Eng. degree in information engineering in 2013. She is currently pursuing the M.S. degree with the School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China.

Her current research interests include network security and privacy protection.

Jiamou Liu received the Ph.D. degree in computer science from the University of Auckland, Auckland, New Zealand, in 2010.

He is a Senior Lecturer with the School of Computer Science, University of Auckland. He was a Senior Lecturer with the Auckland University of Technology, Auckland, from 2011 to 2015 and a Researcher with the Department of Computer Science, Leipzig University, Leipzig, Germany, from 2009 to 2010. His current research interests include social network analysis, multiagent systems, and algorithms.

Keke Gai received the B.Eng. degree in automation from the Nanjing University of Science and Technology, Nanjing, China, in 2004, the M.B.A. degree in business administration from Lawrence Technological University, Southfield, MI, USA, in 2009, the Master of Educational Technology degree in educational technology from the University of British Columbia, Vancouver, BC, Canada, in 2010, the M.S. degree in information technology from Lawrence Technological University in 2014, and the Ph.D. degree in computer science from Pace University, New York, NY, USA.

He is currently an Associate Professor with the School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China. He has published 3 books and over 100 peer-reviewed journal/conference papers. His current research interests include cyber security, edge computing, cloud computing, blockchain, and reinforcement learning.

Dr. Gai was a recipient of five IEEE Best Paper Awards (e.g., TrustCom 18' and HPCC 18') and two IEEE Best Student Paper Awards in recent five years. His research about edge computing was named Best Research Paper of 2018 by the *Journal of Network and Computer Applications*.