AIoT-Empowered Smart Grid Energy Management with Distributed Control and Non-Intrusive Load Monitoring

Linfeng Shen*‡, Feng Wang†, Miao Zhang*, Jiangchuan Liu*, Gaoyang Liu‡, Xiaoyi Fan‡
*School of Computing Science, Simon Fraser University, Canada
†Department of Computer and Information Science, The University of Mississippi, USA
‡Shenzhen Jiangxing Intelligence Inc., Shenzhen, China

2023 IEEE/ACM 31st International Symposium on Quality of Service (IWQoS) | 979-8-3503-9973-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/IWQOS57198.2023.10188781

***Abstract*—Today's electrical grid is experiencing a fast transition toward a smart infrastructure. Modern smart grid is expected to integrate Artificial Intelligence of Things (AIoT)-empowered *energy management systems* (EMS) to sense, analyze, and optimize the power consumption and QoS of diverse end users. Non-Intrusive Load Monitoring (NILM) plays a key role in this transition, particularly considering that many legacy devices/appliances may not have built-in sensors. Yet most of the NILM solutions rely on large (often impractical) datasets for training. In this paper, we address this challenge through a meta learning-inspired approach, which implements a hierarchical architecture with a "meta-learner" to supervise the training of each appliance. Current EMS also relies on a central controller to access long-term information across all participants, which mismatches their distributed nature, and so often with slow responses. To this end, we develop a deep reinforcement learning based controller to make dynamic decisions for each component in the system. The experiment results based on real-world data sets and simulation data show that applying the meta learning approach can greatly improve the performance of NILM and the QoS of the whole system.**

I. Introduction

As the governments of many countries have made commitments to limit the annual carbon emissions and urban waste by 2050, the electrical grid has also started a transition trend from the traditional fossil-based grid to a sustainable and digitalized smart/green grid [1]. One core enabling technology towards this trend is Internet of Things (IoT) and the further Artificial Intelligence of Things (AIoT), providing the increasing availability of sensed data and allowing automated online temporal analyses and thus data-driven solutions [2]. As such, AIoT-Based Smart Grid (SG) is proposed as a significant enhancement of the traditional power grid to better utilize recent advances on information technologies and renewable power generations like wind and solar, with the capabilities to establish bidirectional communications and power flows among consumers and utilities, which can in turn adopt corresponding strategies to maintain the Quality of Service (QoS) for modern smart IoT-enabled energy systems, such as improving transmission efficiency, reducing pollution emissions, improving grid security, responding to disturbances quickly and so many other benefits [3] [4].

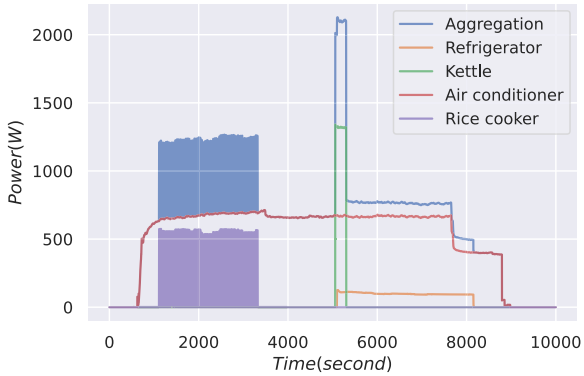


Fig. 1: NILM for appliances in a house.

One critical functionality that lies in the center of SG is the AIoT-empowered energy management system (EMS) for a microgrid. With all the components in the system connected by network and the data collected and analyzed by a central controller, the system can provide more opportunities to increase energy usage efficiency, save cost, improve the security level and so on [5] [6] [7]. For example, an EMS in a residential microgrid can involve renewable power generation units, energy storage systems (ESS) and appliances in one or more houses sharing a nearby neighborhood. A central controller collected the data of all the components in the system and schedule the operation of ESS and appliances to minimize the operation cost while maximizing the comfort level of customers.

One of the key tasks in such an EMS is predicting the operation time of appliances from the historical data and using this to maximize the comfort level of customers. Yet, although more and more appliances in the smart home have equipped with networking modules and can be controlled remotely nowadays, the household penetration of smart appliances is still at a low rate (5.5%) and will not grow to a substantial portion (15%) in the near future [8]. More importantly, many old houses only have one central power meter, and it is not possible to post-install a smart meter for each of the appliances. This makes the EMS can hardly optimize the grid

without enough detailed information about such appliances. Fortunately, Non-Intrusive Load Monitoring (NILM) [9], also known as energy disaggregation, has shown its great potential to be the data "sensor" for these appliances without physical interaction and "connect" them into nowadays IoT-based smart grid, where the operation information of each appliance can be predicted from the central meter and then utilized by the EMS for optimization.

As an example, Fig. 1 shows NILM in a house with several appliances. Specifically, the input of NILM is the aggregated sequence of power consumption which is recorded by a central meter (e.g., a smart meter in a house). The objective of NILM is to estimate the power consumption of each appliance from the input of aggregated power consumption sequence. Different roles in the smart grid can benefit from NILM. For customers, the appliance-level power consumption feedback can help them get more detailed information on power consumption in their house, and it has been shown that providing disaggregated power consumption of appliances can save up to 20% of power per dwelling [10]. For utilities, NILM can be guidelines for demand response [11]. In particular, utility companies can shed specific loads that are less important during peak hours with detailed information on power consumption to relieve the burden of the grid [12]. On the other hand, although recent years deep learning methods have been widely used for NILM [13] [14] [15] [16] [17] [18], most of them still need a large volume of data for training to acceptable results. This is because different appliances have quite unique power consumption patterns and even the same type of appliance can vary depending on its brand. Moreover, in most of the current deep learning based NILM, the model of each appliance is trained independently, which further slows down the learning process and makes it hard for practical deployment.

In this paper, we propose a hierarchical meta learning-inspired approach for NILM to fast learn the patterns of appliances with limited training data and leverage the correlation of different appliances. The key idea is to utilize a tiered architecture to train the models. Besides the prediction model of each appliance which we call the "learner", there is also a model named "meta-learner" which supervises the training of each appliance by generating the parameters of each "learner" model. Different from the previous deep learning methods that need to train a model for each appliance, our method can train one model for the prediction of all the appliances. Our approach can be easily applied to most of existing deep learning based methods such as Seq2Seq and Seq2Point [14]. To maximize its benefits, we further adapt a new deep learning model Conv-TasNet [19] into the NILM scenario and propose a meta learning inspired TasNet NILM (MeTas-NILM) model based on our approach.

In addition, most of the current EMS [6] [7] assume the central controller can acquire all the information of the system in a long period and use these for optimization. The real situation is that the controller can only get limited information and make decisions based on the current observation of the system. To solve this dynamic decision problem, we propose

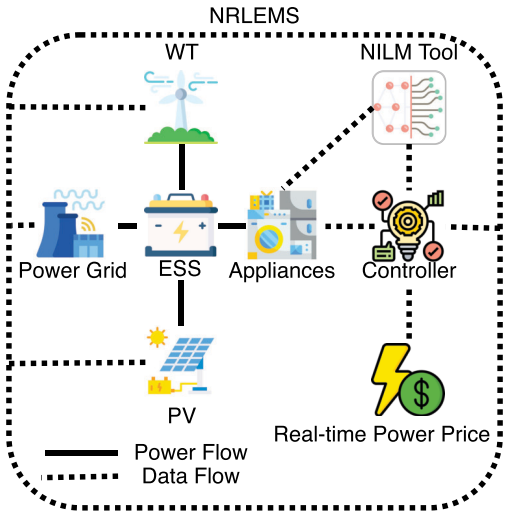


Fig. 2: Framework of NRLEMS

an EMS for an AIoT-empowered residential microgrid based on NILM and deep reinforcement learning (DRL), which we call **NRLEMS**. We use the Actor-Critic method based on *Proximal Policy Optimization (PPO)* [20] algorithm to solve this RL problem.

The results of our extensive experiments based on real-world datasets and simulations on our own generated dataset show that our hierarchical meta learning-inspired approach can greatly improve the accuracy of NILM. And our MeTas-NILM can also achieve fast learning objective by efficient parameter sharing with even better results by appliance-specific parameterization. The performance evaluation shows the effectiveness of the **NRLEMS** and further comparison shows that the better prediction results of NILM improve the Quality of Service (QoS) of the whole system.

II. NRLEMS Design and System Model

In this paper, we consider an energy management system in an AIoT-empowered residential microgrid that has some renewable power generation units, an energy storage system (ESS), and several appliances in one dwelling, as illustrated in Fig. 2. For the power provider side, the power grid is modeled as an infinite power source with real-time power price changing over time. Two types of renewable power generators, photovoltaic (PV) power plants and wind turbines (WT), provide free power for the residential microgrid. Although the current energy harvesting systems can support self-sustainable or energy neutral operations [21], we still consider the power grid for generality. A central controller schedules and operates the appliances and the ESS unit, where the former serve as the consumers in the microgrid, and the latter can be both the providers and consumers. The preferred operation time can be specified by customers manually or predicted by the NILM tool using the historical data. Each component in the system is connected to the network, so as to be controllable by the central controller. The QoS of the system can be

measured as minimizing the operation cost of the microgrid while maximizing the comfort level of the customers, which is determined by the preferred operation time and the true operation time of appliances. The following of this section will describe the mathematical model of each part and the problem definition.

Previous works [6] [22] have shown that ESS can not only serve as backup power during a blackout but also provide opportunities for demand reshaping to save cost in both industrial and residential scenarios. State of Charge (SoC) measures the level of charge of the ESS relative to its capacity. Let t denote the time index in a time period T, and the SoC of ESS in time $t+1$ can be calculated as

$$SoC(t+1) = SoC(t) + S_{ESS}(t) \cdot \eta_{ch(dch)} \cdot R_{ch(dch)} \quad (1)$$

where $\eta_{ch(dch)}$is charging (discharging) efficiency of ESS, $R_{ch(dch)}$ is charging (discharging) rate of ESS, and $S_{ESS}(t)$ is the status of ESS. $S_{ESS}(t) = 1$ if ESS is charging, $S_{ESS}(t) = -1$ if ESS is discharging, and $S_{ESS}(t) = 0$ if ESS is not used. The SoC of ESS should also be limited in a certain lower and upper bound to have a longer lifetime as shown in Eq. 2. The operation cost of ESS at time t $C_{ESS}(t)$ can be represented as Eq. 3, where c_{ESS} is the operation cost of ESS for one time unit.

$$SoC_{min} \leq SoC(t) \leq SoC_{max} \quad (2)$$

$$C_{ESS}(t) = |S_{ESS}(t)| \cdot c_{ESS} \quad (3)$$

Another cost considered in our system is incurred by the power demand from the power grid. Let $P_{app}(t)$ denote the power demand of all the appliances in the microgrid at time t, the power demand from the power grid $P_{grid}(t)$ can be calculated as Eq. 4, where $P_{ESS}(t) = S_{ESS}(t) \cdot \eta_{ch(dch)} \cdot R_{ch(dch)}$ is the power consumed or provided by the ESS at time t. $P_{WT}(t)$ and $P_{PV}(t)$ are the power provided by the WT and PV, which are determined by the weather condition at time t (wind speed and solar radiation).

$$P_{grid}(t) = max\{0, P_{app}(t) + P_{ESS}(t) - P_{WT}(t) - P_{PV}(t)\} \quad (4)$$

The total operation cost OC can be calculated as the summation of money paid for power from the power grid and the cost of ESS. Our first objective is to minimize the total cost of the microgrid, as shown in Eq. 5, where $RTP(t)$ is the real-time price of power at time t.

$$\textbf{Min: } OC = \sum_{t \in T} [RTP(t) \cdot P_{grid}(t) + C_{ESS}(t)] \quad (5)$$

Another objective is to maximize the comfort level of customers, which is measured by the preferred operation time of each appliance and the true operation time scheduled by the system [6]. If the preferred operation time is not specified by the customer, we use the operation time predicted by the NILM tool on historical data as the preferred operation time. For a number of N appliances considered in the system, let OT_n and POT_n denote the true operation time and preferred operation time of the n-th appliance, the total comfort level CL can be calculated as Eq. 6, where ξ is a penalty factor and SD_n is the satisfaction degree of the n-th appliance.

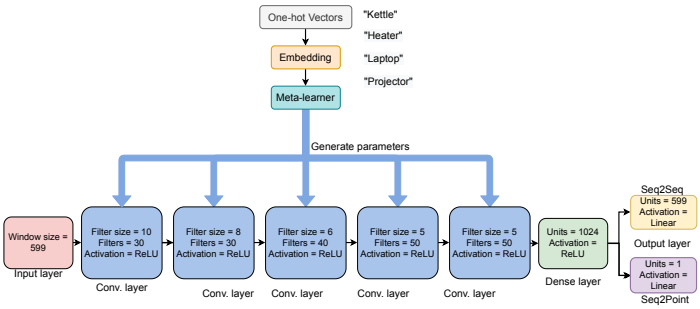


Fig. 3: The architecture of Meta-Seq2Seq and Meta-Seq2Point.

$$\textbf{Max: } CL = \sum_{n \in N} (1 - \xi \cdot |OT_n - POT_n|) \cdot SD_n \quad (6)$$

Based on the above modeling of each part in the system, our objective to minimize the operation cost while maintaining the maximum comfort level of customers (QoS of the whole system) can be represented as Eq. 7, where w_1 and w_2 are two coefficients to adjust the importance of each objective.

$$\textbf{Max: } \Psi = w_1 \cdot CL - w_2 \cdot OC \quad (7)$$

III. Meta Leaning-Inspired Approach for NILM

The NILM tool is a crucial component in **NRLEMS** and its accuracy largely determine the QoS of the system. At first, we will give the specific problem statement of NILM. Let x_t^i denotes the power consumption of i-th appliance at time step t. The aggregated power consumption sequence at time t can be represented as

$$y_t = \sum_{i \in N} x_t^i + \epsilon_t \quad (8)$$

where N is the set of all appliances, and ϵ_t is the Gaussian noise with zero mean and variance σ_t^2 at time t. Let $\mathbf{y} = (y_1, y_2, ..., y_T)$ and $\mathbf{x}^i = (x_1^i, x_2^i, ..., x_T^i)$ denote the power consumption sequence of the aggregation and the i-th appliance over a time period T. The objective of NILM is to estimate each power consumption sequence $\mathbf{x}^i$ from the input of aggregation sequence $\mathbf{y}$. In this section, we will first introduce the baseline methods **Seq2Seq** and **Seq2Point** that we will compare with our approaches in the experiments. Then we will introduce our meta-learning inspired approaches **Meta-Seq2Seq** and **Meta-Seq2Point** extended from the baselines. At last, we will propose a more complicated approach **MeTas-NILM** which leverages Conv-TasNet [19], a deep learning approach for end-to-end time-domain speech separation.

A. Meta-Seq2Seq and Meta-Seq2Point

In previous works [13] [14] [17], deep learning approaches for NILM use sliding windows $\mathbf{y}_{t,w} = (y_t, ..., y_{t+w-1})$ as inputs for practical reasons. In [14], instead of predicting the corresponding sliding window $\mathbf{x}_{t,w} = (x_t, ..., x_{t+w-1})$ of the target appliance, Zhang et al. propose a method that only predict the midpoint element $x_{t+\lfloor w/2 \rfloor}$ of the target appliance as the output. They have shown that sequence-to-point learning outperforms previous sequence-to-sequence work for NILM both in experiments and theoretical analysis. The inputs of the models are the sliding window of the aggregated power sequence and the outputs are the predicted power sequence of a specific appliance. The only difference between these models is that the **Seq2Point** only has the midpoint of the sliding window as the output. However, these models have several problems. At first, they need to train a model for each appliance and it will be very time-consuming. Another problem is that they do not leverage the correlation among different appliances, which means each appliance is trained independently and the model does not consider the similarities and dissimilarities among different appliances.

To this end, we want to leverage the correlation among different appliances and want to train the model quickly. We also start from the **Seq2Seq** and **Seq2Point** baselines models. But instead of directly training the model for each appliance, we use a meta learning-inspired architecture as shown in Fig. 3. We use another model which we call "meta-learner" to generate some parameters of the previous **Seq2Seq** and **Seq2Point** models. The inputs of the "meta-learner" are the one-hot vectors of all the appliances, these vectors are then projected into embedding of the appliances E where the "meta-learner" can encode the attributes of the appliances in multiple dimensions and use these as guidelines to generate the parameters of the model for prediction (the "learner"). The "meta-learner" generates the kernel parameters of convolution layers in the "learner" as:

$$\Theta_k := L_k B_k E \tag{9}$$

where $B_k \in \mathbb{R}^{M' \times M}$ and $L_k \in \mathbb{R}^{|\Theta_k| \times M'}$ are learnable linear functions that generate the kernel parameters of the k-th convolution layer Θ_k. M is the dimension of the original embedding E and M' is the bottleneck dimension which is further constrained with $M' < M$ so that the "meta-learner" extracts the M' most relevant characteristics of the appliances. The architecture of "learner" remains same for all the appliances. In this way, we only need to train one model for all the appliances while still enabling appliance-specific parameterization. In the experiments, we found this can help to save about half of the time. And the "meta-learner" can also encode the attributes – similarities and dissimilarities of different appliances, which are expected to improve the prediction results of the appliances.

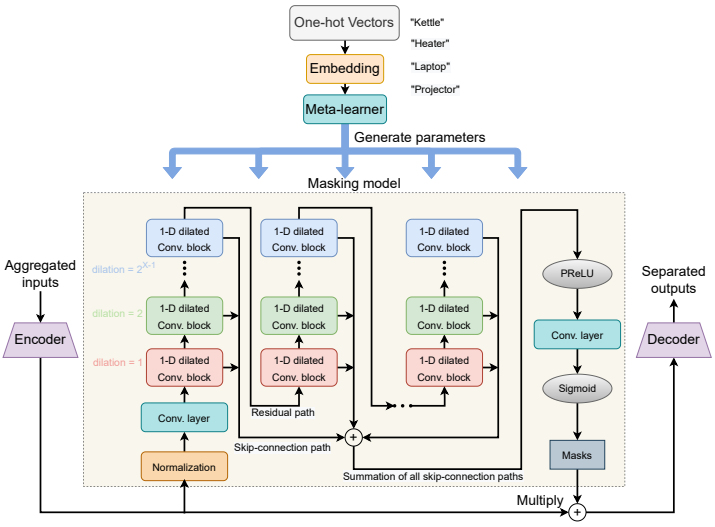


Fig. 4: The architecture of MeTas-NILM.

B. MeTas-NILM

Another problem of the baseline models is that they have relatively simple architecture and do not consider the data sequence with different time range, only with several simple convolution layers and linear layers. We also apply our ideas to a more complicated model and want to see if this can improve the performance. We propose another meta learning-inspired approach based on *Conv-TasNet* [19] which we call **MeTas-NILM**. The architecture of **MeTas-NILM** is shown in Fig. 4. The "meta-learner" part is same as the **Meta-Seq2Seq** and **Meta-Seq2Point** while the "learner" part is based on Conv-TasNet [19]. The "learner" has three sub-components to implement the tasks of NILM. At first, given a sliding window of aggregated power consumption of all appliances $\mathbf{y}_{t,w} = (y_t, ..., y_{t+w-1})$, the encoder encodes them into a latent high-dimensional representation of power consumption signals $r = encoder(\mathbf{y}_{t,w}) \in \mathbb{R}^{l \times w}$ where l is the dimension of the latent representation. Then this latent representation will be the inputs of a masking model which calculates the masks of each appliance $m_i = mask_i(r) \in [0, 1]^{l \times w}$. The separated representation of each appliance $\hat{r_i}$ is obtained by the element-wise product of each appliance's masks and the representation of the aggregation: $\hat{r_i} = r \odot m_i$. In this way, the model is expected to identify the targeted area of each appliance in the representation domain. At last, a decoder (with the same architecture as the encode) reconstructs the separated power consumption sequence of each appliance $\mathbf{x}^i_{t,w} = (x^i_t, ..., x^i_{t+w-1})$ from the masked representation: $\mathbf{x}^i_{t,w} = decoder(\hat{r_i}) \in \mathbb{R}^w$.

For the encoder and decoder in the "learner", we use a more complicated architecture instead of a single 1-D convolutional layer used in the original TasNet. The inputs of the aggregated power signal are passed to n convolutional layers with a different number of filters and different filter sizes. In specific, the $i-$th convolutional layer has $1/4^i \times L$ filters and the size of each filter is $2^i \times W$, where L and W are the base number of

filters and base size of each filter. Then the information of all n convolutional layers is concatenated together and passed to two additional 1-D convolutional layers to generate the latent representation $r \in \mathbb{R}^{l \times w}$. In this way, our encoder is capable of capturing the features from the aggregated inputs in the different time ranges. The decoder uses a similar architecture as the encoder to match the capacity of the encoder. The masked representation of each appliance $\hat{r_i} \in \mathbb{R}^{l \times w}$ is first transformed by two Conv-ReLU layers. Then the outputs are split and passed to n transposed 1-D convolutional layers with the same number and size of filters as in the encoder. At last, the outputs of all layers and summed together to construct the estimated power consumption sequence of each appliance $\mathbf{x}^i_{t,w} = (x^i_t, ..., x^i_{t+w-1})$.

In the architecture of the "learner", the masking model is of most interest since it contains the appliance-specific information and the encoder and decoder are appliance-agnostic and remain the same for the whole task. So the "meta-learner" model in **MeTas-NILM** will generate the parameters of the k-th convolutional layers in the masking model Φ_k in a similar way to **Meta-Seq2Seq** and **Meta-Seq2Point**:

$$\Phi_k := L_k B_k E \tag{10}$$

We use the fully-convolutional separation model proposed in [19] as the masking model in **MeTas-NILM**. This model is based on the temporal convolutional network (TCN) [23] [24], which is proposed as a replacement for RNNs in sequence modeling tasks. As shown in Fig. 4, the masking model consists of several layers of stacked 1-D dilated convolution blocks. The dilation factors of each block increase exponentially to get large temporal context information considering the different range dependencies of the power signal, as denoted with different colors in Fig. 4. In the masking model, X convolution blocks with $1, 2, ..., 2^{X-1}$ dilation factors are repeated H times. To ensure the output length is the same as the input length, the input to each block is zero-padded accordingly. Then the outputs of all the blocks are added together and passed to a convolution block with kernel size 1 for mask estimation.

Compared with **Meta-Seq2Seq** and **Meta-Seq2Point**, **MeTas-NILM** is more complicated and have more parameters. It takes a long time to train, but we expect it to have better performance. We will validate this hypothesis in the performance evaluation.

IV. DRL-driven Central Controller

The central controller in our system needs to dynamically schedule the operation of each appliance and the ESS unit given the input of real-time price of power, weather condition, and the customer pattern predicted by the NILM tool. As deep reinforcement learning (DRL) has been shown to achieve great success in solving similar sequential decision problems with dynamic environments in other research areas, we propose a DRL-driven central controller design in this section to further improve the overall performance of our system. To this end, we first convert the central controller scheduling problem into an RL task and then use a policy gradient algorithm named Actor-Critic method with *Proximal Policy Optimization (PPO)* [20] algorithm to solve it. Since the state space in our problem is high-dimensional, we use Neural Networks (NNs) to represent both the actor and critic components.

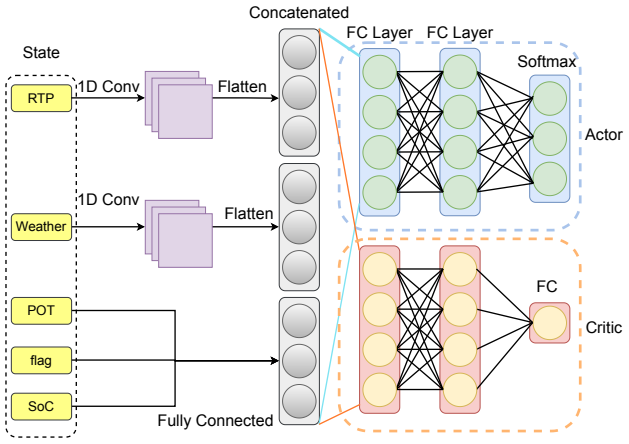


Fig. 5: Network architecture

At first, we define the state of the environment at each time step t. According to the modeling of the environment in Section II, the state s_t can be represented as

$$s_t = (\overrightarrow{RTP}, \overrightarrow{weather}, \overrightarrow{POT}, \overrightarrow{flag}, SoC) \tag{11}$$

where $\overrightarrow{RTP}$ and $\overrightarrow{weather}$ are vectors to represent the real-time price of power and weather conditions in a time period. $\overrightarrow{POT}$ is a vector to represent the preferred operation time of each appliance, either specified by the customer or predicted by the NILM tool. $\overrightarrow{flag}$ is a vector to show if each appliance has already been operated recently. At last, the scalar SoC is the current State of Charge of ESS. Given the state and action of each time step, the environment gives back the reward r_t which is calculated as Eq. 7.

The network architecture of the actor and critic is shown in Fig. 5. For time-series input like RTP and weather conditions, we use 1D convolution layers. For all the other scalar input we use fully connected layers. Then all the results are flattened and concatenated to a new layer, which is then fed to both the actor and critic network. For each input state s_t, the *actor* component needs to output an action a_t which determines the schedule of each appliance and the ESS. There are three status for the ESS, charge, discharge or not use. So we use two bits to represent the operation of the ESS. Similarly, for each appliance, there are two status, close or open. So we use one bit to represent the operation of each appliance. To represent all the operation combinations, we need a binary with $2 + N$ bits to represent the action. Fig. 6 shows an example action representation with 4 appliances. The first two bits are "10" to represent that the ESS is charging at time t. The latter four bits show the status of each appliance. In this example, "0010" means that only the second appliance is open at time t.

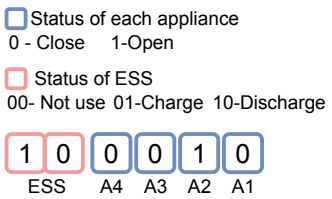


Fig. 6: Example of action representation

Unfortunately, the standard policy gradient algorithm can have great turbulence while training, which means the network may fall into a dead end and can never recover. To this end, we utilize the *Proximal Policy Optimization (PPO)* [20] algorithm to clip the size of each training step. Instead of the traditional objective $L_t^P(\theta) = \hat{\mathbb{E}}_t[\log \pi_\theta(a_t|s_t)\hat{A}_t]$, *PPO* uses the clipped surrogate objective as

$$L_t^{clip}(\theta) = \hat{\mathbb{E}}_t[min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t] \quad (12)$$

where

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \quad (13)$$

$r_t(\theta)$ is the probability ratio of the new and old policy. $\hat{A}_t$ is the estimate of the *advantage* function and ϵ controls the clip bounds. At last, the total loss function is the summation of this clipped *PPO* objective and two additional terms:

$$L_t^{total}(\theta) = \hat{\mathbb{E}}_t[L_t^{clip}(\theta) - c_1 MSE(V(s)) + c_2 S[\pi_\theta](s_t)] \quad (14)$$

where c_1 and c_2 are coefficients of the two terms. The first term is the mean square error of the value function $V(s)$, which is used to update the baseline network. The second term is the entropy of the policy, which is used to ensure enough exploration during training. Since we use two distinct networks for *actor* and *critic* components, we do not need this second term in our problem.

Let α denote the learning rate, and the policy parameter θ can be updated as the gradient ascent of the total loss:

$$\theta \longleftarrow \theta + \alpha \sum_t \nabla_\theta L_t^{total}(\theta) \quad (15)$$

V. Performance Evaluation

In this section, we first introduce the real-world data sets, the simulation data sets generated by ourselves, and the evaluation metrics. We then present the evaluation results of different NILM models. At last, we will evaluate the QoS of the whole **NRLEMS**. All of the models are trained on NVIDIA RTX A5000 and implemented in Python using PyTorch.

A. Data Sets and Evaluation Metrics

To evaluate the performance of **MeTas-NILM**, we use both experiments and simulations. There are several real-world data sets such as **REDD** [25], **UK-DALE** [26] and **REFIT** [27]. We use **UK-DALE** for experiments. In **UK-DALE**, both the aggregated and individual appliance power consumption were recorded every 6 seconds. The data set contains the information of 5 UK houses from November 2012 to April 2017. We normalize the data using the same way as the previous work [14].

In the evaluation, we use two metrics to compare different methods. The first one is the normalized signal aggregate error (SAE). This metric is useful when we are interested in the total error of power consumption over a period. Let $\hat{x}_t$ and x_t denote the ground truth and the prediction of an appliance power consumption at time t. The metric SAE can be represented as

$$SAE = \frac{|\sum_t \hat{x}_t - \sum_t x_t|}{\sum_t \hat{x}_t} \quad (16)$$

The second metric is the mean absolute error (MAE). This metric is useful when we are interested in the average error of power consumption over a period T. The metric MAE can be represented as

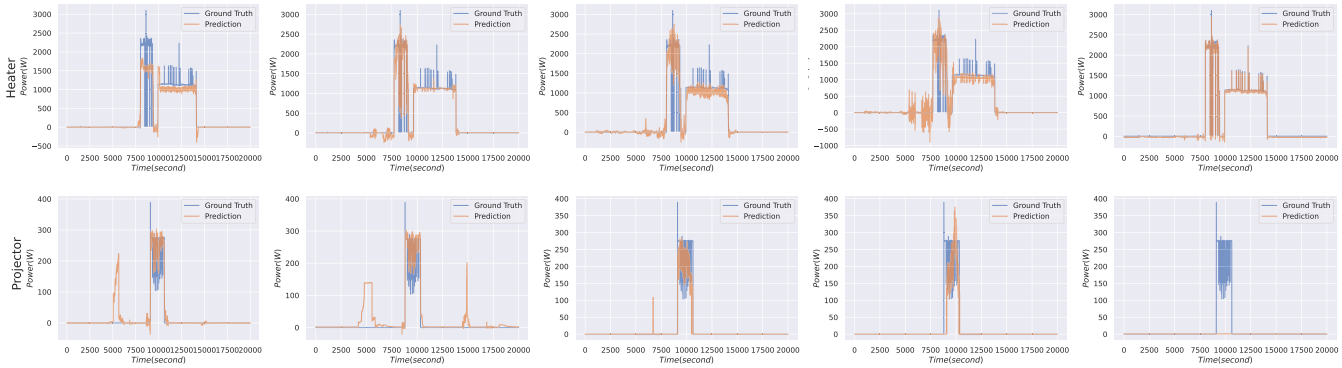
$$MAE = \frac{1}{T}\sum_{t=1}^{T} |\hat{x}_t - x_t| \quad (17)$$

B. Results of MeTas-NILM

We first compared **Meta-Seq2Seq**, **Meta-Seq2Point** and **MeTas-NILM** with **Seq2Seq** and **Seq2Point** using the data of House3 in **UK-DALE** data set. House3 has 4 types of appliances: kettle, heater, laptop, and projector. There are about 510,000 rows of data in total (about 36 days of data in 6 seconds intervals). We use 80% of the data for training and use the remaining 20% for testing.

Fig. 7 shows the plots of the prediction results of different approaches. We can see that meta learning-based approaches can predict the results better. Our **MeTas-NILM** can have better prediction results in most of the cases but still miss some representational patterns of certain appliances. For example, as shown in Fig. 7-(e), **MeTas-NILM** does not predict the pattern of the Projector appliance well. Table. I shows the comparison of SAE and MAE metrics on different appliances in House3. The results show that although the improvement for the SAE metric is not very significant since the baselines already get small SAE values, our meta-learning approaches can remarkably improve the prediction results compared to the baselines for the MAE metrics. In particular, **Meta-Seq2Seq** decreases the value from 17.90 to 17.22 compared to **Seq2Seq** and **Meta-Seq2Point** decreases the value from 19.44 to 18.04 compared to **Seq2Point**. Our **MeTas-NILM** approach can reduce the mean error by 39% compared to **Seq2Seq**.

As the time interval of the real world data set is long (from 3 seconds to 8 seconds), we also collected some data from different appliances in one second time interval. The types of appliances we have are air conditioner, refrigerator, hairdryer, rice cooker, television, drum washing machine, oven, air heater, kettle, hot-water heater, microwave, turbo washing machine, vacuum cleaner, and range hoods. For appliances



(a) Seq2Seq (b) Seq2Point (c) Meta-Seq2Seq (d) Meta-Seq2Point (e) Meta-NILM

Fig. 7: Prediction results of different appliances in UK-DALE House3 from different methods

TABLE I: Comparison of normalized signal aggregate error (SAE) and mean absolute error (MAE) on different appliances in UK-DALE House3

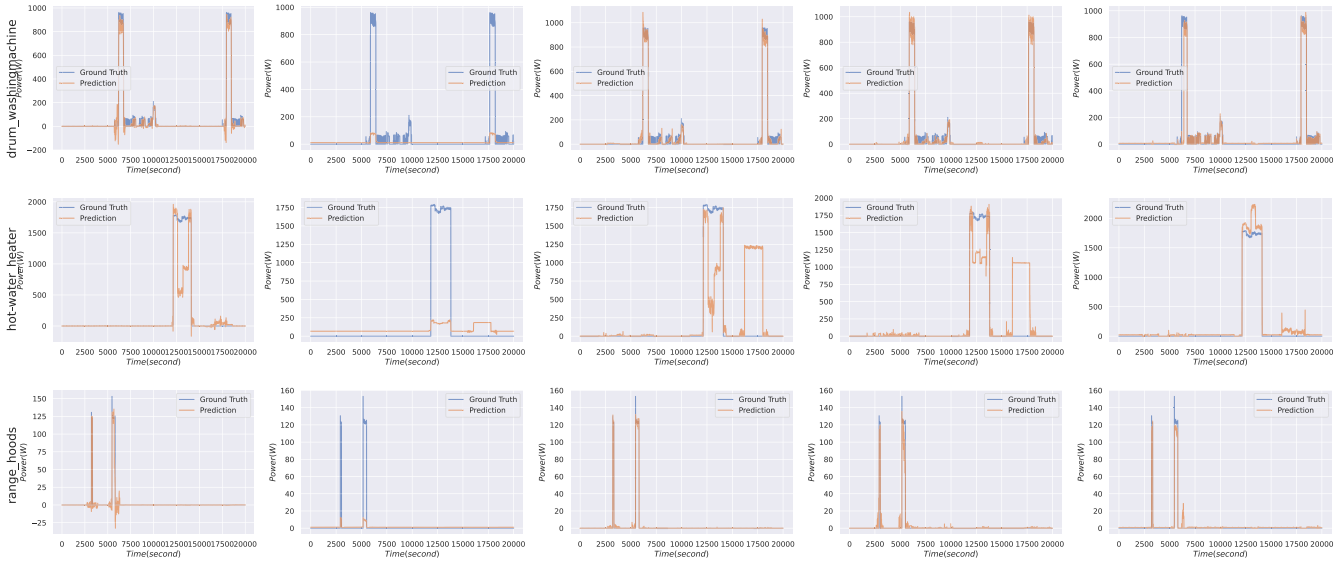
Error measures	appliances	Seq2Seq	Seq2Point	Meta-Seq2Seq	Meta-Seq2Point	Meta-NILM
SAE	Kettle	0.11	0.14	0.15	0.15	**0.08**
	Heater	0.11	**0.02**	0.17	0.14	**0.02**
	Laptop	0.77	2.17	0.74	0.88	**0.70**
	Projector	0.14	0.20	**0.09**	**0.09**	0.17
	Overall	0.28±0.28	0.63±0.89	0.89±0.26	0.32±0.33	**0.24±0.26**
MAE	Kettle	5.47	14.56	13.58	13.58	**4.94**
	Heater	23.82	13.91	18.66	15.02	**7.36**
	Laptop	40.35	44.16	34.90	41.16	**24.82**
	Projector	2.60	5.11	**2.18**	2.43	6.80
	Overall	17.90±15.17	19.44±14.75	17.22±11.77	18.04±14.21	**10.98±8.04**

that have different working modes (such as hairdryers and air heaters), we also collected data of the appliance in the different working states. As our data set only records the data in a short time period, we generate large-scale data with data augmentation. For the appliances that only have one working mode (e.g., kettle), we generate the simulated data with random time intervals among each operating period of the appliance. For the appliances that have several working modes, we generate the simulated data by randomly selecting one working mode, and then adding random time intervals among each operating period of the appliance. The aggregation power consumption sequence is generated by adding the power consumption data of all the appliances. In this way, our simulation data set can include the combination of different appliances in a random pattern. We call the generated data set as **Simulation** in the rest of the paper.

We generate 1,000,000 rows of data (about 12 days of data in one-second intervals) for evaluation. We use 800,000 rows of data for training and 200,000 rows of data for testing. Fig. 8 shows the plots of the prediction results for different appliances from different approaches. To save space we choose three appliances to display: drum washing machine, hot-water heater, and range hoods. The results show that our **Meta-NILM** can have better prediction results in most of the cases. We also compare the MAE and SAE metrics on different appliances in **Simulation** data set. The results show that our meta-learning approaches can improve the prediction results compared to the original approaches. For the MAE metrics, **Meta-Seq2Seq** decreases the value from 15.38 to 10.23 compared to **Seq2Seq** and **Meta-Seq2Point** decreases the value from 20.36 to 11.34 compared to **Seq2Point**. Our **MeTas-NILM** approach can achieve the best performance with about 50% improvements of MAE value and about 34% improvements of SAE value compared to **Seq2Seq**.

The evaluation results of both experiments on real-world data sets and simulation show that meta learning-inspired approaches can achieve better performance compared to the baselines for the NILM problem. Then to evaluate if meta learning-inspired approaches can train the model faster compared to the baselines that train each appliance independently, we also compare the training time of different approaches on each data set. The results show that the original **Seq2Seq** and **Seq2Point** models need about 1 hour to train one model for each appliance and the total time to train all the appliances is about 4 hours. But for the meta learning-based models. We can train all the four appliances in about 2 hours, which can save about half of the time. **MeTas-NILM** needs a much longer time to train because of its complicated architecture. But the training time of **MeTas-NILM** is still less than training all the appliances using baseline approaches.

The experiment and simulation results we get show that our meta-learning inspired approaches can train the model faster and better with a relatively small data set. With such a hierarchical architecture inspired by meta learning, our



(a) Seq2Seq (b) Seq2Point (c) Meta-Seq2Seq (d) Meta-Seq2Point (e) Meta-NILM

Fig. 8: Prediction results of different appliances in Simulation data set from different approaches

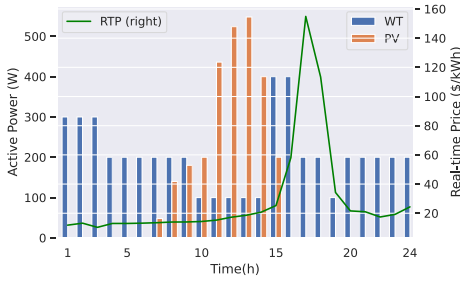


Fig. 9: Data example of one day

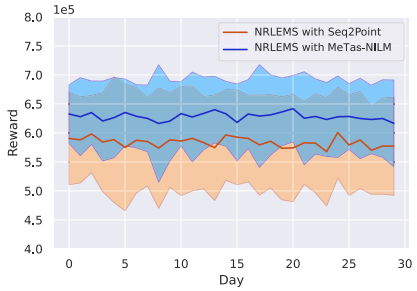


Fig. 10: Results of NRLEMS

approach can have more parameter sharing in the "learner" part which can facilitate the training process. On the other hand, the "meta-learner" part encodes the correlation among different appliances and use this information to generate some of the parameters in the "learner". This can help the models maintain appliance-specific information in the training process and leverage the commonalities of different appliances, which improves the prediction results of each appliance.

C. Performance Evaluation of NRLEMS

At last, we evaluate the performance of the proposed NRLEMS on the **Simulation** data set. The RTP information is from ENGIE Resources [28], and the weather conditions (wind speed and solar radiation) are from NSRDB: National Solar Radiation Database [29]. Fig. 9 shows the RTP and PV, WT generation information in one day. According to the weather condition, PV and WT generation can have drastic turbulence. The RTP in one day can also vary as shown by the green line in Fig. 9. All of these factors make the scheduling of the appliances and ESS unit a big challenge. To this end, we utilize DRL to train our central controller. We denote 30 days as one episode in our environment. We train our system for 5000 episodes and test it for 1000 episodes. The results of our NRLMES with **Seq2Point** and **MeTas-NILM** are shown in Fig. 10. Our **MeTas-NILM** method can improve the performance of NRLEMS for about 9% compared to **Seq2Point**.

VI. Related Work

A. Applications of NILM in Smart Grid

NILM technology enables getting the appliance-level behavior of the customers using only the aggregation data (which is usually recorded by one smart meter). It helps to mitigate the requirements such as high-cost sensors on each appliance, which is not realistic in most cases. NILM provides a cost-efficient solution to get detailed power consumption information of appliances and can be applied in different situations. For residential microgrids, NILM can get the information of user behavior and be guidelines for appliances operation schedules. For example, Çimen et al. [6] propose an efficient NILM-based energy management system

(EMS) for residential microgrids. The detailed results of different appliances obtained by NILM can be integrated and analyzed in an EMS to create an efficient and user-centered microgrid operation schedule. The simulation results show that the proposed NILM-based EMS can save the overall cost while improving the customer satisfaction ratio compared to a traditional EMS. For industrial users who are the main load for the grid, it is not possible for centralized monitoring of power consumption due to the wide geography distribution like data centers. NILM can be integrated into the distributed system of each data center and analyze the relationship between the workload and the power consumption. For example, Lee et al. [30] propose a power measurement platform for data centers based on NILM. The experiment results show that this platform can recognize the peak power of each node and provide opportunities for informed peak power reduction. Yu et al. [31] adopt NILM on the collaborative computing of the edge devices and edge data centers. By performing the computing tasks on the computing resources close to the data source of industrial users, the pressure of centralized data center, the transmission bandwidth, and security privacy risk are all improved.

B. Traditional Method Based on FHMM

Since Hart first proposed the problem of NILM [9], the factorial hidden Markov model (FHMM) has been the most popular method for a long time. Hidden Markov Models have been shown well in speech recognition and other sequential models. Several works applied FHMM and its variants to tackle NILM problem [32] [33] [34]. The biggest problem of these methods based on FHMM is their high computation complexity, which will increase exponentially with the number of appliances. On the other hand, some of these methods need other information besides the aggregated sequence and each appliance needs to be modeled in detail. All of these factors limit the NILM method based on FHMM, especially for more complex tasks. To this end, most of the recent works turn to deep neural networks (DNN). DNN has shown remarkable results in sequential models, such as speech recognition and translation. Kelly et al. [13] show that applying DNN for NILM achieves better performance than traditional methods based on FHMM.

C. Deep Learning-Based NILM

DNN has been applied in the NILM field since Kelly et al. [13] applied 3 different DNN models on NILM and showed its dominance compared to previous methods. In the following years, various models have been applied to the NILM problem. For example, Zhang et al. [14] propose a sequence-to-point learning method and Shin et al. [17] propose a subtask gated network for NILM. Harell et al. [15] also apply a modified WaveNet model for NILM. These methods however have significant limits in that they need to train a model for each of the appliances. That means we need to collect large volume data of each appliance for training. This has become more and more unrealistic since the types of appliances are increasing rapidly nowadays. There are also some works that only need one model to disaggregate the power consumption jointly. Bejarano et al. [35] propose deep latent generative model. This model can get the appliance-level power consumption with one single model based on variational recurrent neural networks (VRNNs). We only need to train one model but we have to retrain the model every time when new appliances are added to get the new patterns. Extra information such as the consumers' location has also been exploited to get better results [16]. These extra data on the other hand make the training phase more and more complex and unrealistic, which makes the method further and further from quick adaptation to new appliances.

Meta-learning, also known as "learning to learn" [36], aims to design models that can adapt to new tasks rapidly with a few training examples. Prior meta-learning works mainly focus on few-shot classification problem [37] [38], where a model must adapt to new classes not seen in the training phase with limited examples of each new class. Meta-learning can also be applied to different tasks, such as supervised learning, reinforcement learning and other more specific problems. There are also some works that applied meta-learning to source separation problems. Samuel et al. [39] propose a hierarchical model based on meta-learning for music source separation. The experiment results showed that their model contains fewer parameters and runs faster than baselines.

VII. Conclusion and Future work

In this paper, we proposed an energy management system **NRLEMS** in AIoT-empowered smart grid, which utilizes the NILM tool as a data sensor in the system and connects legacy appliances with IoT networks. To improve the performance of NILM, we further proposed meta learning-inspired approaches. We started from two baseline models and extended them to meta learning-inspired approaches. Then we proposed another more complicated meta learning approach **MeTas-NILM** based on TasNet. According to the problem of current EMS, we utilize DRL for the central controller in the AIoT-empowered microgrid to make dynamic decisions. We also collected data from different appliances and generated our own simulation data set by data augmentation. The experiment and simulation results showed that applying the meta learning approach for NILM can improve the performance with a shorter training time. We also evaluated our **NRLEMS** on the simulation data set, and the results show that the improved NILM tool can help to get better QoS of the whole system. Our system is only based on a AIoT-empowerd residential microgrid now. In our future work, we will leverage our approaches on more different scenarios like industrial scenarios. We will also combine more approaches such as federated learning and meta reinforcement learning to improve the performance of **NRLEMS** in the future.

Acknowledgments

Feng Wang's research is partly supported by an NSF I/UCRC Grant (1822104).

REFERENCES

[1] X. Fang, S. Misra, G. Xue, and D. Yang, "Smart grid—The new and improved power grid: A survey," *IEEE communications surveys & tutorials*, vol. 14, no. 4, pp. 944–980, 2011.

[2] A. Gal, A. Senderovich, and M. Weidlich, "Online temporal analysis of complex systems using iot data sensing," in *Proc. of IEEE International Conference on Data Engineering*, 2018.

[3] N. Framework, "Roadmap for smart grid interoperability standards," *National Institute of Standards and Technology*, vol. 26, 2010.

[4] F. Mwasilu, J. J. Justo, E.-K. Kim, T. D. Do, and J.-W. Jung, "Electric vehicles and smart grid interaction: A review on vehicle to grid and renewable energy sources integration," *Renewable and sustainable energy reviews*, vol. 34, pp. 501–516, 2014.

[5] W. Li, T. Logenthiran, V.-T. Phan, and W. L. Woo, "Implemented IoT-based self-learning home management system (SHMS) for Singapore," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2212–2219, 2018.

[6] H. Çimen, N. Çetinkaya, J. C. Vasquez, and J. M. Guerrero, "A microgrid energy management system based on non-intrusive load monitoring via multitask learning," *IEEE Transactions on Smart Grid*, vol. 12, no. 2, pp. 977–987, 2020.

[7] W. Li, T. Logenthiran, V.-T. Phan, and W. L. Woo, "A novel smart energy theft system (SETS) for IoT-based smart home," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5531–5539, 2019.

[8] Smart Appliances. Accessed: 2022-06-30. [Online]. Available: https://www.statista.com/outlook/dmo/smart-home/smart-appliances/worldwide

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

[10] E. Aydin, D. Brounen, and N. Kok, "Information provision and energy consumption: Evidence from a field experiment," *Energy Economics*, vol. 71, pp. 403–410, 2018.

[11] J. Froehlich, E. Larson, S. Gupta, G. Cohn, M. Reynolds, and S. Patel, "Disaggregated end-use energy sensing for the smart grid," *IEEE pervasive computing*, vol. 10, no. 1, pp. 28–39, 2010.

[12] Z. Zhou, F. Liu, Z. Li, and H. Jin, "When smart grid meets geo-distributed cloud: An auction approach to datacenter demand response," in *Proc. of IEEE International Conference on Computer Communications*, 2015.

[13] J. Kelly and W. Knottenbelt, "Neural nilm: Deep neural networks applied to energy disaggregation," in *Proc. of ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, 2015.

[14] C. Zhang, M. Zhong, Z. Wang, N. Goddard, and C. Sutton, "Sequence-to-point learning with neural networks for non-intrusive load monitoring," in *Proc. of the AAAI Conference on Artificial Intelligence*, 2018.

[15] A. Harell, S. Makonin, and I. V. Bajić, "Wavenilm: A causal neural network for power disaggregation from the complex power signal," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019.

[16] C.-Y. Hsu, A. Zeitoun, G.-H. Lee, D. Katabi, and T. Jaakkola, "Self-supervised learning of appliance usage," in *International Conference on Learning Representations*, 2019.

[17] C. Shin, S. Joo, J. Yim, H. Lee, T. Moon, and W. Rhee, "Subtask gated networks for non-intrusive load monitoring," in *Proc. of the AAAI Conference on Artificial Intelligence*, 2019.

[18] X. Wang, H. Zhou, N. M. Freris, W. Zhou, X. Guo, Z. Liu, Y. Ji, and X.-Y. Li, "Lcl: Light contactless low-delay load monitoring via compressive attentional multi-label learning," in *Proc. IEEE/ACM International Symposium on Quality of Service*, 2021.

[19] Y. Luo and N. Mesgarani, "Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation," *Proc. of IEEE/ACM transactions on audio, speech, and language processing*, 2019.

[20] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[21] W. Zhou, H. Zhou, W. Hua, F. Zhou, X. Cui, S. Tang, Z. Liu, and X.-Y. Li, "Imp: Impedance matching enhanced power-delivered-to-load optimization for magnetic mimo wireless power transfer system," in *Proc. IEEE/ACM International Symposium on Quality of Service*, 2021.

[22] G. Tang, H. Yuan, D. Guo, K. Wu, and Y. Wang, "Reusing backup batteries as BESS for power demand reshaping in 5g and beyond," in *Proc. of IEEE International Conference on Computer Communications*, 2021.

[23] C. Lea, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks: A unified approach to action segmentation," in *Proc. of Springer European Conference on Computer Vision*, 2016.

[24] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[25] J. Z. Kolter and M. J. Johnson, "REDD: A public data set for energy disaggregation research," in *Workshop on data mining applications in sustainability*, 2011.

[26] J. Kelly and W. Knottenbelt, "The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes," *Scientific data*, vol. 2, no. 1, pp. 1–14, 2015.

[27] D. Murray, L. Stankovic, and V. Stankovic, "An electrical load measurements dataset of United Kingdom households from a two-year longitudinal study," *Scientific data*, vol. 4, no. 1, pp. 1–12, 2017.

[28] ENGIE Resources. Accessed: 2022-06-30. [Online]. Available: https://www.engieresources.com

[29] NSRDB: National Solar Radiation Database. Accessed: 2022-06-30. [Online]. Available: https://nsrdb.nrel.gov/

[30] S. Lee, H. Kim, S. Park, S. Kim, H. Choe, and S. Yoon, "Cloudsocket: fine-grained power sensing system for datacenters," *IEEE Access*, vol. 6, pp. 49 601–49 610, 2018.

[31] J. Yu, W. Liu, and X. Wu, "Noninvasive industrial power load monitoring based on collaboration of edge device and edge data center," in *Proc. of IEEE International Conference on Edge Computing*, 2020.

[32] H. Kim, M. Marwah, M. Arlitt, G. Lyon, and J. Han, "Unsupervised disaggregation of low frequency power measurements," in *Proc. of the SIAM International Conference on Data Mining*, 2011.

[33] J. Z. Kolter and T. Jaakkola, "Approximate inference in additive factorial hmms with application to energy disaggregation," in *Artificial intelligence and statistics*, 2012.

[34] O. Parson, S. Ghosh, M. Weal, and A. Rogers, "Non-intrusive load monitoring using prior models of general appliance types," in *Proc. of the AAAI Conference on Artificial Intelligence*, 2012.

[35] G. Bejarano, D. DeFazio, and A. Ramesh, "Deep latent generative models for energy disaggregation," in *Proc. of the AAAI Conference on Artificial Intelligence*, 2019.

[36] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[37] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. of International Conference on Machine Learning*, 2017.

[38] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. of Advances in neural information processing systems*, 2017.

[39] D. Samuel, A. Ganeshan, and J. Naradowsky, "Meta-learning extractors for music source separation," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020.