Subtask Gated Networks for Non-Intrusive Load Monitoring

Changho Shin,[1] Sunghwan Joo,[2] Jaeryun Yim,[1] Hyoseop Lee,[1] Taesup Moon,[2] Wonjong Rhee[3]

[1]Encored Technologies, Seoul, Korea
[2]Department of Electrical and Computer Engineering, Sungkyunkwan University, Suwon, Korea
[3]Department of Transdisciplinary Studies, Seoul National University, Seoul, Korea
[1]{chshin, yim, hslee}@encoredtech.com, [2]{shjoo840, tsmoon}@skku.edu, [3]wrhee@snu.ac.kr

Abstract

Non-intrusive load monitoring (NILM), also known as energy disaggregation, is a blind source separation problem where a household's aggregate electricity consumption is broken down into electricity usages of individual appliances. In this way, the cost and trouble of installing many measurement devices over numerous household appliances can be avoided, and only one device needs to be installed. The problem has been well-known since Hart's seminal paper in 1992, and recently significant performance improvements have been achieved by adopting deep networks. In this work, we focus on the idea that appliances have on/off states, and develop a deep network for further performance improvements. Specifically, we propose a subtask gated network that combines the main regression network with an on/off classification subtask network. Unlike typical multitask learning algorithms where multiple tasks simply share the network parameters to take advantage of the relevance among tasks, the subtask gated network multiply the main network's regression output with the subtask's classification probability. When standby-power is additionally learned, the proposed solution surpasses the state-of-the-art performance for most of the benchmark cases. The subtask gated network can be very effective for any problem that inherently has on/off states.

Introduction

Non-intrusive load monitoring (NILM) was first proposed by Hart in 1992, and it is the process of estimating the power consumptions of individual appliances by using the aggregated electricity consumption as the only input (Hart 1992). Because the power consumption of multiple appliances are added together to form the aggregated consumption, and because the goal of NILM is to disaggregate each appliance's power consumption from the aggregate power consumption measured with a single sensor, NILM is also called energy disaggregation. The disaggregated energy consumption information can be used to provide feedback to the consumers and affect their energy consumption behaviors, and for instance Neenan et al. showed that 15% energy saving can be achieved (Neenan, Robinson, and Boisvert 2009). Furthermore, NILM can be used for detecting malfunctioning appliances, designing energy incentives, and managing demand-response (Froehlich et al. 2010). As an exemplary use case,

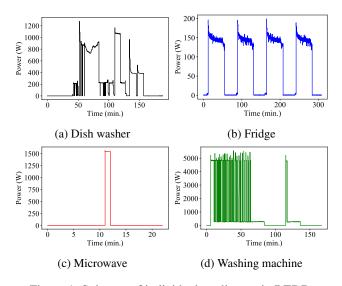(a) Dish washer (b) Fridge (c) Microwave (d) Washing machine

Figure 1: Snippets of individual appliances in REDD.

utility companies can identify which homes are running a particular appliance (e.g., air conditioner) during the peak hours and take proper actions to reduce the power usage.

In the last few years, deep learning has become a popular approach for NILM (Kelly and Knottenbelt 2015; Huss 2015; Zhang et al. 2018). These works show improved NILM performance by applying deep learning algorithms that mainly developed in other fields. However, the previous works do not explicitly exploit the inherent properties of electricity consumption data. In our work, we utilize one of these properties, the on/off state concept of appliances that we can observe in Figure 1. We present subtask gated networks (SGN) that adopt one DNN for regression, and the other DNN for on/off classification. By multiplying regression output with classification probability to form the final estimates, this model outputs the power estimation gated with on/off classification. Through the loss function that explicitly reflects the classification errors, classification subnetwork serves as the on/off based gate. SGN shows 15-30% improved performance in REDD and UK-DALE on average. Also, we further investigated its variants that reflects standby power and hard gating, which shows additional im-

provements for some appliances.

Related Works

Non-intrusive load monitoring

The most frequently used approach for NILM is the Factorial Hidden Markov Model (FHMM) (Ghahramani and Jordan 1997) and its variants (Kim et al. 2011; Kolter and Jaakkola 2012; Parson et al. 2012; Zhong, Goddard, and Sutton 2014; Shaloudegi et al. 2016). Kim et al. proposed an energy disaggregation algorithm based on a conditional factorial hidden semi-Markov model, which exploits additional features related to when and how appliances are used (Kim et al. 2011). Kolter and Jaakkola presented an approximate factorial hidden Markov model based on convex optimization for inference, that aggregates the difference signal and other constraints for aggregated energy data (Kolter and Jaakkola 2012). Parson et al. showed that prior models of appliances in FHMM could be applied to real-world energy disaggregation system (Parson et al. 2012). Zhong et al. incorporated signal aggregate constraints (SACs) into an additive factorial hidden Markov model, which significantly improves the original additive factorial hidden Markov model (Zhong, Goddard, and Sutton 2014). Shaloudegi et al. enhanced Zhong et al.'s algorithms by combining convex semidefinite relaxations randomized rounding and a scalable ADMM optimization algorithm (Shaloudegi et al. 2016).

The latest approaches are based on deep learning. Jack Kelly et al. tried a variety of deep learning models including Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Denoising Autoencoder (DAE) (Kelly and Knottenbelt 2015). Huss proposed a hybrid energy disaggregation algorithm based on CNNs and a hidden semi-Markov model (Huss 2015). Zhang et al. suggested a sequence-to-point learning with CNN such that the single mid-point of a time window is treated as the output of the network instead of the whole sequence of the window (Zhang et al. 2018). While showing significant performance improvements, none of the approaches utilize on/off states of appliances with end-to-end training.

Multi-task learning

Our work was inspired by Multi-Task Learning (MTL). In general, multi-task learning learns with multiple tasks that are associated with task-specific loss functions (Ruder 2017). Multi-task learning methods enable models to generalize better for the original task by sharing parameters among the related tasks. Caruana summarizes the goal of MTL as following — "MTL improves generalization by leveraging the domain-specific information contained in the training signals of related tasks" (Caruana 1998).

Most multitask learning studies have focused on improving the performance of the main task or all tasks by lower layer parameter sharing among multiple tasks (Zhang et al. 2014; Yu and Jiang 2016; Liu et al. 2015; Cheng, Fang, and Ostendorf 2015). For example, In the work of Zhang et al., head pose estimation and facial attribute inference tasks are used as the auxiliary tasks, which shares parameters except for the last layer's parameters that participate in each tasks'

output. These related tasks improve the facial landmark detection task by learning better parameter values (Zhang et al. 2014).

The proposed architecture in this paper, subtask gated networks, can be interpreted as multi-task learning with an auxiliary task of classification. However, there is a major difference in the way subtask output is used. In our work, the auxiliary task output is used as a multiplication unit to calculate the final output of the main task, and this is beyond parameter sharings that indirectly affect the main task through the loss functions.

Gating mechanism

Gating mechanism in deep learning means softly selecting one of two or more components using sigmoid or softmax function. For example, sigmoid has been widely used to determine whether a hidden state should be memorized or not in recurrent neural networks (Hochreiter and Schmidhuber 1997; Gers and Schmidhuber 2000; Cho et al. 2014). As another example, in the mixture of experts (Jacobs et al. 1991), softmax has been used to decide which expert to use for each input region. However, in these examples, gating networks cannot explicitly learn from gating task itself. Instead, these examples use EM algorithm or backpropagation with the loss function related with the final output, not the gating itself. In our subtask gated network, we added a sort of 'gating loss' using auxiliary task label that is available, which makes our model learn from on/off gating task directly.

Preliminary

Problem formulation of energy disaggregation

Given the aggregate power consumption for time T periods as $\mathbf{x} = (x_1, x_2, \cdots, x_T)$, let $\mathbf{y}^i = (y_1^i, y_2^i, \cdots, y_T^i)$ denote the power consumption of i-th appliance in the house. At each time step, the aggregate power consumption could be represented as the summation of individual power consumptions as follows

$$x_t = \sum_i y_t^i + \epsilon_t, \tag{1}$$

where ϵ_t is assumed Gaussian noise with zero mean and variance σ_t^2. Suppose we are only interested in I appliances, which are widely used household appliances in most of households. Then, the power consumption from unknown appliances can be represented $\mathbf{u} = (u_1, u_2, \cdots, u_T)$, and (1) can be reformulated as follows

$$x_t = \sum_i^I y_t^i + u_t + \epsilon_t. \tag{2}$$

The disaggregation problem is then to estimate the power consumption sequences of appliances $\mathbf{y}^1, \mathbf{y}^2, \cdots, \mathbf{y}^I$ from $\mathbf{x}$.

Deep neural networks for energy disaggregation

In previous studies (Kelly and Knottenbelt 2015; Zhang et al. 2018), deep neural networks for nonlinear regression are used to estimate the power consumption of individual appliances from the sequence of aggregated power
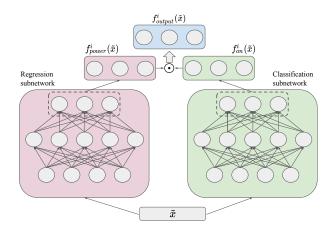
Figure 2: A generic subtask gated network.

consumptions. For practical reasons, deep neural networks use partial sequences $\mathbf{x}_{t,s} := (x_t, \cdots, x_{t+s-1})$ and $\mathbf{y}_{t,s}^i := (y_t^i, \cdots, y_{t+s-1}^i)$ starting at t with length s as input and output respectively, rather than the whole sequences of $\mathbf{x}$ and $\mathbf{y}^i$. To avoid the loss of context information, we can consider additional windows of length w on both end sides, only for input. To be precise, $\tilde{\mathbf{x}}_{t,s,w} := (x_{t-w}, \cdots, x_{t+s+w-1})$ as the input and $\mathbf{y}_{t,s}^i$ as the output. Since the sequence length s and the additional window length w are fixed throughout this paper, we will omit the subscripts s and w on notations for simplicity. In our work, we denote the appliance power estimation model $f_{power}^i : \mathbb{R}_+^{s+2w} \to \mathbb{R}_+^s$ for an individual appliance is defined as

$$\hat{\mathbf{p}}_t^i = f_{power}^i(\tilde{\mathbf{x}}_t). \tag{3}$$

Architectures Although RNN architectures are widely used for sequence modeling (Sutskever, Vinyals, and Le 2014), CNN is also an attractive solution for energy disaggregation problems and shows better performance than the RNN architectures (Kelly and Knottenbelt 2015; Huss 2015; Zhang et al. 2018). Recently, an empirical study showed that CNNs can outperform RNNs across a diverse range of sequence modeling tasks and data sets (Bai, Kolter, and Koltun 2018). When some of RNNs were tried for our problem, they showed worse performance than CNNs even with a longer training time. Therefore, RNNs are excluded in this study and we only consider the CNN model used in Zhang's paper (Zhang et al. 2018).

Subtask Gated Networks (SGN)

Auxiliary classification subtask for NILM

Our general framework uses the auxiliary classification subtask as the gating unit of the main regression task as shown in Figure 2. Using the subtask, we can exploit the on/off state in an explicit way. To be precise, we can formulate on/off classification as a subtask for non-intrusive load monitoring.

Let $\mathbf{o}^i = (o_1^i, o_2^i, \cdots, o_T^i)$ be the state sequence of appliance i where $o_t^i \in \{0, 1\}$ represents the on/off state of appliance i at time t:

$$o_t^i = \begin{cases} 1, & \text{if } y_t^i > \text{threshold}, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Again, we use the notation $\mathbf{o}_{t,s}^i := (o_t^i, \cdots, o_{t+s-1}^i)$ for partial sequence. The subscript s will be omitted as long as the sequence length s is fixed. And then, we define a neural network $f_{on}^i : \mathbb{R}_+^{s+2w} \to [0,1]^s$ for on/off state classification of appliance i, which is the mapping:

$$\hat{\mathbf{o}}_t^i = f_{on}^i(\tilde{\mathbf{x}}_t). \tag{5}$$

We want to emphasize that each element in $\hat{\mathbf{o}}_t^i$ means a probability that an appliance has 'on' state.

We define f_{output}^i, the final output of our structure, as

$$f_{output}^i(\tilde{\mathbf{x}}_t) = f_{power}^i(\tilde{\mathbf{x}}_t) \odot f_{on}^i(\tilde{\mathbf{x}}_t) \tag{6}$$

where $\odot$ represents the element-wise multiplication. Then, this architecture naturally induces the following loss functions:

$$\mathcal{L}_{output}^i = \frac{1}{T} \sum_{t=1}^{T} (y_t^i - \hat{p}_t^i \hat{o}_t^i)^2, \tag{7a}$$

$$\mathcal{L}_{power}^i = \frac{1}{T} \sum_{t=1}^{T} (y_t^i - \hat{p}_t^i)^2, \tag{7b}$$

$$\mathcal{L}_{on}^i = -\frac{1}{T} \sum_{t=1}^{T} \left(o_t^i \log \hat{o}_t^i + \left(1 - o_t^i\right) \log(1 - \hat{o}_t^i) \right). \tag{7c}$$

Note that $\mathcal{L}_{output}^i$ and $\mathcal{L}_{power}^i$ are the mean squared error loss in the whole network and the regression subnetwork for the power estimation task, respectively. $\mathcal{L}_{power}^i$ is the loss function used in literatures (Kelly and Knottenbelt 2015; Zhang et al. 2018). $\mathcal{L}_{on}^i$ is the sigmoid cross entropy loss in the classification subnetwork.

Loss function

In our work, we use the following loss function for the joint optimization.

$$\mathcal{L} = \mathcal{L}_{output}^i + \mathcal{L}_{on}^i. \tag{8}$$

This loss function represents the sum of the overall network loss and the classification subnetwork loss. Our model can be trained using $\mathcal{L}_{output}^i$ only, but, $\mathcal{L}_{on}^i$ directly instill on/off state information to the classification subnetwork. It induces the classification subnetwork to operate as an on/off gate and using $\mathcal{L}_{output}^i$ and $\mathcal{L}_{on}^i$ shows better performance.

The loss function and its gradient for the whole network are:

$$\mathcal{L}_{output}^i = \frac{1}{T} \sum_{t}^{T} (y_t^i - \hat{p}_t^i \hat{o}_t^i)^2,$$

$$\frac{\partial \mathcal{L}_{output}^i}{\partial \hat{y}_j^i} = -\frac{2}{T} \hat{o}_j^i \left(y_j^i - \hat{p}_j^i \hat{o}_j^i \right),$$

$$\frac{\partial \mathcal{L}_{output}^i}{\partial \hat{o}_j^i} = -\frac{2}{T} \hat{y}_j^i \left(y_j^i - \hat{p}_j^i \hat{o}_j^i \right).$$

Above equations show that the regression subnetwork learns only when the classification subnetwork classifies the state of the target appliance as 'on' state. However, if the classification subnetwork is saturated as zero for the input, the power estimation subnetwork cannot learn because gradients are also zeros. For that case, minimizing $\partial \mathcal{L}_{output}^i / \partial \hat{o}_j^i$ can revive the classification network later if the regression subnetwork outputs non-zero values.

Variants of subtask gated networks for NILM

Here, we consider several options to enhance the performance of SGN for NILM problem.

	REDD	UK-DALE
Dish washer	0.27	0.61
Fridge	3.82	3.02
Kettle	-	1.10
Microwave	3.66	0.50
Washing machine	1.14	2.76

Table 1: Mean standby power (W) of appliances.

SGN with standby power (SGN-sp) Appliances may have non-zero power consumption even when it is in 'off' state. By analyzing data, we have easily identified standby power as shown in Table 1. They may seem small, but they are not negligible because the 'off' state takes up a significant portion of the time. Because SGN has the assumption that an appliance power consumption is zero when it is in 'off' state, a slight modification is required to take into account standby power. Considering that standby power is a fixed value when the appliance is in 'off' state, the formulation can be simply modified as follows.

$$f_{output}^i(\tilde{\mathbf{x}}_t) = f_{power}^i(\tilde{\mathbf{x}}_t) \odot f_{on}^i(\tilde{\mathbf{x}}_t) + (\mathbb{1} - f_{on}^i(\tilde{\mathbf{x}}_t))b, \quad (9)$$

where b is a learnable scalar.

Hard SGN Instead of multiplying the power estimation value by the probability output of classification subnetwork as it is, multiplying the power estimation by 0 or 1 depending on the output of classification subnetwork may be more compliant to the meaning of the on/off gating. The hard gating can be made simply by applying a condition function $g(x)$ such that

$$g(x) = \begin{cases} 1, & \text{if } x \geq 0.5, \\ 0, & \text{otherwise.} \end{cases}$$

The final output is modified as follows.

$$f_{output}^i(\tilde{\mathbf{x}}_t) = f_{power}^i(\tilde{\mathbf{x}}_t) \odot g(f_{on}^i(\tilde{\mathbf{x}}_t)). \quad (10)$$

We can also derive **Hard SGN-sp** model straightforwardly with the modified output formulation as follows.

$$f_{output}^i(\tilde{\mathbf{x}}_t) = f_{power}^i(\tilde{\mathbf{x}}_t) \odot g(f_{on}^i(\tilde{\mathbf{x}}_t)) + (\mathbb{1} - g(f_{on}^i(\tilde{\mathbf{x}}_t)))b. \quad (11)$$

Note that $\mathcal{L}_{on}$ is still based on $\hat{o}_t^i$ in Hard SGN and Hard SGN-sp, not $g(\hat{o}_t^i)$.

Experiments

Experiment settings

Datasets We evaluate our proposed methods on the two real-world datasets, REDD (Kolter and Johnson 2011) and UK-DALE (Kelly and Knottenbelt 2014). The REDD dataset has the data for six US houses, and the UK-DALE dataset has the electricity usage data for five UK houses respectively.

In REDD, the aggregate power consumptions were recorded in every 1 second, and the appliance power consumptions were recorded in every 3 seconds over various durations. The dataset contains the measurements of the aggregate power consumption and 10-25 types of appliances. However, we only consider microwave, fridge, dish washer and washing machine as in the previous work (Zhang et al. 2018). We used the data of house 2–6 as the training set, and house 1 as the test set.

In UK-DALE, all data were recorded in every 6 seconds from November 2012 to January 2015. The dataset contains the main aggregated power consumption and measurements of 4-54 appliances. We only consider kettle, microwave, fridge, dish washer and washing machine in our experiments. For evaluation, we used house 1 and 3–5 for training, and house 2 for testing as in the previous work (Zhang et al. 2018). We only used the last week data that was published after preprocessing[1].

Data Preprocessing Since the UK-DALE data set has been preprocessed by Jack Kelly (Kelly and Knottenbelt 2015), we used it as it is. For REDD dataset, we preprocessed with the following procedure to handle missing values. First, we split the sequence so that the duration of missing values in subsequence is less than 20 seconds. Second, we filled the missing values in each subsequence by the backward filling method. Finally, we only used the subsequences with more than one-day duration. For both UK-DALE and REDD data, the aggregate power consumptions and appliances' power consumptions were divided by standard deviations of the aggregate power consumptions.

Neural Net Training Details As a baseline, the performance of FHMM was evaluated using FHMM implementations in NILMTK, an open source toolkit for analysis on energy disaggregation (Batra et al. 2014). Jack Kelly's denoising autoencoder (DAE) and Zhang's CNN (Seq2Seq) were implemented according to the architectures described in each paper (Kelly and Knottenbelt 2015; Zhang et al. 2018) for evaluations. Our model architecture is shown in Figure 3. This architecture uses Zhang's Seq2Seq as the two subnetworks because neural network architecture itself is not the main focus of our work. More details of each architecture can be found in the supplementary document. Our model has the following hyperparameters. The learning rate is 1.0×10^{-4}, and the batch size is 16. The label for the on/off classification task is generated using 15 watts as the threshold according to equation (4). The DNN models are trained on NVIDIA GTX 1080Ti and implemented using TensorFlow 1.8 package. He initialization (He et al. 2015) is used for the weights of all neural network architectures.

---

[1]http://jack-kelly.com/files/neuralnilm/NeuralNILM_data.zip

Data was sliced with additional window size w=400 and output sequence length s=64 for REDD, w=200 and s=32 for UK-DALE, in which the input is a sequence of 43.2 minutes and the output is a sequence of 3.2 minutes. We used Adam optimizer (Kingma and Ba 2015) for training.
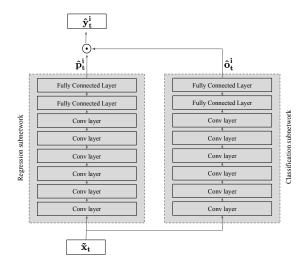

Figure 3: SGN architecture used in our experiments.

Evaluation Metrics In this experiment, MAE (Mean Absolute Error) and SAE_δ (Signal Aggregate Error per time δ period) were used to evaluate the performance. MAE is a general metric used for regression problems, and it shows how well models perform power consumption estimation in NILM. The SAE_δ represents the average total error over the time period of δ, which compare the sum of ground truth and estimation over the time period of δ. We use the time normalized version of signal aggregate error (SAE). Unlike the original SAE in previous works (Zhang et al. 2018) that considers the power consumption of overall period by summing up, we compare the usage per hour, since this may be a more important concept regarding energy consumption

behavior control[2]. Note that SAE_δ is normalized using the time period (an hour), not the ground truth power consumption as the previous works (Zhang et al. 2018). Although it makes an apple-to-apple comparison difficult among different appliances, it can prevent outliers caused by close-to-zero denominator and division by zero when the appliance is in the 'off' state.

Let y_t^i and $\hat{y}_t^i$ be the ground truth and the estimated power for appliance i at time t, respectively. The MAE for appliance i is defined as: $MAE^i = \frac{1}{T}\sum_{t=1}^{T}|y_t^i - \hat{y}_t^i|$. SAE_δ for appliance i is defined as: $SAE_\delta^i = \frac{1}{T_\delta}\sum_{\tau=1}^{T_\delta}\frac{1}{N_\delta}|r_\tau^i - \hat{r}_\tau^i|$, where N_δ is the number of data points in the time period δ, T_δ is the total time period in time δ scale, and $T = T_\delta \times N_\delta$. r_τ and $\hat{r}_\tau$ represents the sum of power consumption in a given period, $r_\tau^i = \sum_{t=1}^{N_\delta} y_{N_\delta\tau+t}^i$, and $\hat{r}_\tau^i = \sum_{t=1}^{N_\delta} \hat{y}_{N_\delta\tau+t}^i$. In our experiments, N_δ is 1200 for REDD, and 600 for UK-DALE, which correspond to the number of data points in an hour.

Experiment results

Table 2 and 3 show the performance of the previous works and SGN for the REDD and UK-DALE dataset. In the two tables, the bold font denotes the best performing algorithm for each appliance, and the shade denotes that SGN has outperformed the best of the previous works. For MAE, we can see that the performance is improved for 9 out of 9 cases using SGN-sp, and 7 out of 9 cases using SGN. For SAE, the performance is improved for 6 out of 9 cases using either SGN or SGN-sp. On the average, SGN and SGN-sp show 15-30% lower error performance compared to the state-of-the-art. These performance improvements were achieved without fine tuning any of the network architecture, hyper-parameters, or the weight between $\mathcal{L}_{output}$ and $\mathcal{L}_{on}$. Hard SGNs tend to perform worse than SGNs on the average but still perform better than the previous works, and significant

---

[2]We also use various values of δ to analyze time period sensitivity on evaluation (Figure 7).

Metric	Model	Dish washer	Fridge	Microwave	Washing machine	Average	Average improvement
	FHMM (Batra et al. 2014)	101.30	98.67	87.00	66.76	88.43	-
	DAE (Kelly and Knottenbelt 2015)	29.38	76.62	21.31	31.35	39.66	-
	Seq2Seq (Zhang et al. 2018)	27.07	26.03	16.57	22.72	23.10	0.00 %
MAE	SGN	**14.97**	23.89	17.52	20.07	19.09	17.34 %
	SGN-sp	15.96	22.89	**15.98**	20.61	**18.86**	18.36 %
	Hard SGN	21.27	24.45	17.38	**18.24**	20.33	11.97 %
	Hard SGN-sp	24.29	**22.86**	17.16	21.94	21.56	6.67 %
	FHMM	93.64	46.73	65.03	58.77	66.04	-
	DAE	29.21	20.48	17.86	27.64	23.80	-
	Seq2Seq	26.93	11.67	11.43	16.82	16.71	0.00 %
SAE_δ	SGN	**11.74**	**10.62**	14.84	10.70	**11.97**	28.34 %
	SGN-sp	12.07	12.26	**11.31**	13.28	12.23	26.81 %
	Hard SGN	20.72	14.51	16.53	**8.47**	12.40	25.77 %
	Hard SGN-sp	27.63	10.99	15.52	15.66	13.31	20.37 %

Table 2: Experiment results for REDD data.

Metric	Model	Dish washer	Fridge	Kettle	Microwave	Washing machine	Average	Average improvement
MAE	FHMM	48.25	60.93	38.02	43.63	67.91	51.75	-
	DAE	27.42	19.93	12.53	16.77	18.30	18.99	-
	Seq2Seq	23.16	16.71	11.07	9.87	11.94	14.55	0.00 %
	SGN	15.50	15.79	7.08	6.26	12.31	11.39	21.72 %
	SGN-sp	**13.49**	15.32	**5.96**	8.64	**11.04**	**10.89**	25.15 %
	Hard SGN	15.74	16.08	6.27	**4.53**	12.40	11.00	24.38 %
	Hard SGN-sp	15.15	**10.93**	7.40	19.27	13.31	13.21	9.21 %
SAE_δ	FHMM	46.04	51.90	35.41	41.52	64.15	47.80	-
	DAE	23.10	5.99	7.78	14.58	14.36	13.16	-
	Seq2Seq	17.31	**3.74**	6.85	8.98	**9.90**	9.36	0.00 %
	SGN	10.03	7.10	4.81	5.99	10.40	7.66	18.09 %
	SGN-sp	**8.80**	7.56	**3.99**	7.90	10.28	7.71	17.63 %
	Hard SGN	10.54	7.21	4.35	**4.45**	11.18	**7.55**	19.34 %
	Hard SGN-sp	9.04	3.75	5.21	17.52	13.11	9.73	-3.95 %

Table 3: Experiment results for UK-DALE data.

improvements were observed for fridges and microwaves. In the real world, we do not need to apply a single model to all the appliances. We can choose from SGN and its variants depending on the appliances type.

Figure 4 shows three examples of how the classification subnetwork interacts with the regression subnetwork. In the figure, 'SGN before gating' represents $\hat{p}_t^i$ and 'SGN after gating' represents $\hat{p}_t^i \cdot \hat{o}_t^i$ respectively. In the case of the dish washer, gating mainly works for filtering out small noisy regression estimates when the dish washer is really in the 'off' state. In the case of the fridge, the regression subnetwork estimates the power consumption of the 'on' state even when the fridge is clearly in 'off' state. The classification subnetwork is fully responsible for deciding whether it is 'on' or 'off', and the final output after gating turns out to be quite accurate. In the case of the kettle, regression subnetwork seems to have noisy outputs for 'off' state just as in the dish washer case. However, the noise is not small valued as in the dish washer. When other appliances are turned on, the kettle's regression subnetwork output seems to be strongly affected and result in a large valued output. However, again the classification subnetwork does its job of gating, and the final output becomes accurate.

Figure 5 shows the histogram of $\hat{o}_t^i$, the output of the classification subnetwork. Note that the y-axis is in log scale to properly visualize the large counts around zero and one. Although the main goal is a regression task, a large portion of $\hat{o}_t^i$ values is concentrated near zero and one. This is because $\mathcal{L}_{on}$ is included in the cost function in addition to $\mathcal{L}_{output}$. Including $\mathcal{L}_{on}$ affects how the regression subnetwork and classification subnetwork cooperate. To illustrate this point, we have generated the same figure as in Figure 4(b), but this time without including $\mathcal{L}_{on}$. The results are shown in Figure 6. Because $\mathcal{L}_{on}$ is not explicitly included in the cost function, there is no need to perform the classification task well, and the subgated network is free to decide what it wants to do with the two subnetworks. In Figure 6(a), it can be seen that the classification subnetwork became a scaling factor where regression subnetwork simply over-estimates for the 'on' states. When the 'on' probability histogram is compared as shown in Figure 6(b), it is clear that the classification subnetwork has failed to do what it was intended to do. By adding $\mathcal{L}_{on}$ to the cost function, we can prevent the subgated network from being confused and therefore achieve better performance. Going back to Figure 5, it can be seen that the 'on' probability reflects how the appliances really function.
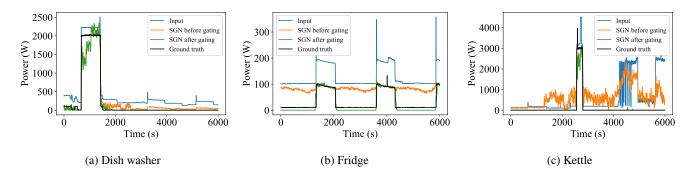

(a) Dish washer

(b) Fridge

(c) Kettle

Figure 4: Gating examples (UK-DALE).
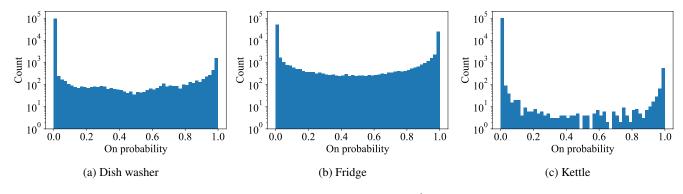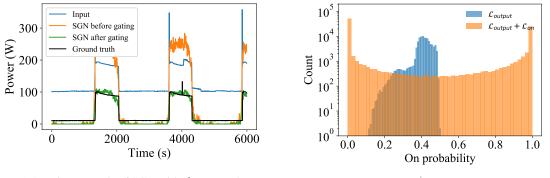
(a) Dish washer

(b) Fridge

(c) Kettle

Figure 5: Histograms of 'on' probability $\hat{o}_t^i$ (UK-DALE).

(a) A gating example of SGN with $\mathcal{L}_{output}$ only

(b) 'On' probability ($\hat{o}_t^i$) histogram comparison

Figure 6: Gating example of UK-DALE fridge. Depending on the choice of loss function, the classification subnetwork's behavior changes.
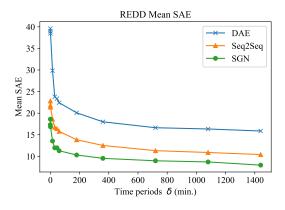

Figure 7: SAE_δ as a function of δ (REDD).

For the fridge that cycles between 'on' and 'off', roughly the same counts can be seen near probability zero and probability one. On the other hand, the counts are skewed for dish washer and kettle that are known to be off most of the time.

δ sensitivity analysis on SAE_δ In practice, NILM needs to be used in different ways. Sometimes each appliance's energy usage per hour is needed for the application, but sometimes usage over a day or a month might be needed as well.

Depending on the application's requirement, SAE's δ needs to be chosen appropriately. To understand if SGN works well for different choices of δ, we have plotted SAE_δ as a function of δ in Figure 7. As expected, the performance becomes better as δ increases because positive and negative errors can have a chance to cancel out. SGN is the best performing algorithm for the entire range of δ that was examined.

Conclusions

In this work, we have proposed subtask gated networks that are effective for the regression problems with on/off states. For the application of non-intrusive load monitoring, performance was evaluated for REDD and UK-DALE datasets. Our proposed solution performed 15-30% better than the state-of-the-art deep learning solution when MAE and SAE_δ were used as the metrics. The results indicate that combining regression subnetwork and classification subnetwork is a promising direction for improving the performance of regression tasks that has on/off state inherently.

Future research may be extended in two possible directions. The first is to generalize how to reflect the appliance state information to the model. Our method handles only the on/off states of appliances, but some appliances have multiple states. It might be possible to use the information by creating precise label information beyond on/off states. The

second is to extend our method to the data of other domains that have the property similar to on/off. Our method can be applied to any regression problems where we can generate an auxiliary class describing discrete states related to the output.

Acknowledgments

References

Bai, S.; Kolter, J. Z.; and Koltun, V. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Batra, N.; Kelly, J.; Parson, O.; Dutta, H.; Knottenbelt, W.; Rogers, A.; Singh, A.; and Srivastava, M. 2014. Nilmtk: an open source toolkit for non-intrusive load monitoring. In *Proceedings of the 5th international conference on Future energy systems*, 265–276. ACM.

Caruana, R. 1998. Multitask learning. In *Learning to learn*. Springer. 95–133.

Cheng, H.; Fang, H.; and Ostendorf, M. 2015. Open-domain name error detection using a multi-task rnn. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 737–746.

Cho, K.; Van Merriënboer, B.; Bahdanau, D.; and Bengio, Y. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Froehlich, J.; Larson, E.; Gupta, S.; Cohn, G.; Reynolds, M. S.; and Patel, S. N. 2010. Disaggregated end-use energy sensing for the smart grid. *IEEE Pervasive Computing* (1):28–39.

Gers, F. A., and Schmidhuber, J. 2000. Recurrent nets that time and count. In *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*, volume 3, 189–194. IEEE.

Ghahramani, Z., and Jordan, M. I. 1997. Factorial hidden markov models. *Machine learning* 29(2-3):245–273.

Hart, G. W. 1992. Nonintrusive appliance load monitoring. *Proceedings of the IEEE* 80(12):1870–1891.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, 1026–1034.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Huss, A. 2015. Hybrid model approach to appliance load disaggregation: Expressive appliance modelling by combining convolutional neural networks and hidden semi markov models.

Jacobs, R. A.; Jordan, M. I.; Nowlan, S. J.; and Hinton, G. E. 1991. Adaptive mixtures of local experts. *Neural computation* 3(1):79–87.

Kelly, J., and Knottenbelt, W. 2014. Uk-dale: A dataset recording uk domestic appliance-level electricity demand and whole-house demand. *ArXiv e-prints* 59.

Kelly, J., and Knottenbelt, W. 2015. Neural nilm: Deep neural networks applied to energy disaggregation. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, 55–64. ACM.

Kim, H.; Marwah, M.; Arlitt, M. F.; Lyon, G.; and Han, J. 2011. Unsupervised disaggregation of low frequency power measurements. In *SDM*, volume 11, 747–758. SIAM.

Kingma, D. P., and Ba, L. 2015. Adam: A method for stochastic optimization.

Kolter, J. Z., and Jaakkola, T. 2012. Approximate inference in additive factorial hmms with application to energy disaggregation. In *International conference on artificial intelligence and statistics*, 1472–1482.

Kolter, J. Z., and Johnson, M. J. 2011. Redd: A public data set for energy disaggregation research. In *Workshop on Data Mining Applications in Sustainability (SIGKDD), San Diego, CA*, volume 25, 59–62. Citeseer.

Liu, X.; Gao, J.; He, X.; Deng, L.; Duh, K.; and Wang, Y.-Y. 2015. Representation learning using multi-task deep neural networks for semantic classification and information retrieval.

Neenan, B.; Robinson, J.; and Boisvert, R. 2009. Residential electricity use feedback: A research synthesis and economic framework. *Electric Power Research Institute*.

Parson, O.; Ghosh, S.; Weal, M.; and Rogers, A. 2012. Nonintrusive load monitoring using prior models of general appliance types.

Ruder, S. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Shaloudegi, K.; György, A.; Szepesvari, C.; and Xu, W. 2016. Sdp relaxation with randomized rounding for energy disaggregation. In *Advances in Neural Information Processing Systems*, 4978–4986.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.

Yu, J., and Jiang, J. 2016. Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 236–246.

Zhang, Z.; Luo, P.; Loy, C. C.; and Tang, X. 2014. Facial landmark detection by deep multi-task learning. In *European Conference on Computer Vision*, 94–108. Springer.

Zhang, C.; Zhong, M.; Wang, Z.; Goddard, N.; and Sutton, C. 2018. Sequence-to-point learning with neural networks for nonintrusive load monitoring. *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*.

Zhong, M.; Goddard, N.; and Sutton, C. 2014. Signal aggregate constraints in additive factorial hmms, with application to energy disaggregation. In *Advances in Neural Information Processing Systems*, 3590–3598.