Available online at www.sciencedirect.com

ScienceDirect

Energy Procedia 158 (2019) 2683–2688

Energy Procedia

www.elsevier.com/locate/procedia

10th International Conference on Applied Energy (ICAE2018), 22-25 August 2018, Hong Kong, China

Neural Network with Extended Input for Estimating Electricity Consumption Using Background-based Data Generation

Gaochen Cui[a], Bo Liu[a], and Wenpeng Luan[a,b,*]

[a]*Key Laboratory of Smart Grid of Ministry of Education, Tianjin University, Tianjin 300072, China*
[b]*China Electric Power Research Institute, Beijing 100192, China*

Abstract

Non-intrusive load monitoring (NILM) can infer the electricity consumption of individual target appliances by only collecting and analyzing the aggregate power data at the single power entrance point. This detailed information obtained is of great significance to smart grid operation and energy saving. For solving the problems of feature loss and training data acquisition in current deep neural network-based NILM methods, this paper presents an extended input neural network method using background-based data generation for estimating electricity consumption. First, the extended input method is proposed, which ensures a complete working cycle to be fed into the neural network each time. Then, a synthetic method based on the power use background is used to generate abundant and desirable training data. Finally, the proposed methods are validated through comparison tests.

Keywords: Non-intrusive load monitoring; Electricity consumption; Deep neural network; Synthetic data

1. Introduction

Non-intrusive load monitoring (NILM) [1] can infer the power use information of the target appliances downstream by only using one sensor installed at the power entrance point to collect the aggregate power data of the house. The detailed appliance-level electricity consumption information obtained by NILM is of great significance to smart grid planning and management, demand-side response, building electricity saving, and energy efficiency improvement [2].

* Corresponding author. Tel.: +86-10-82813446; fax: +86-10-82813323.
E-mail address: luanwenpeng@epri.sgcc.com.cn

1876-6102
10.1016/j.egypro.2019.02.022

Traditionally, there are two basic kinds of methods for achieving NILM. Event-based methods [3-6] detect and classify the on/off events of the appliances to track their working states. They are usually restricted to the appliances with distinguishable state changes, and need manual feature selection. Optimization-based methods [7-9] seek the greatest possible combination of all appliances by solving an optimization problem. They need priori knowledge of every appliance in the house, and the time complexity increases exponentially with the number of appliances.

Recently, deep neural network (DNN) has been widely applied in the fields of artificial intelligence. In NILM, researches in [10,11] have successfully trained a DNN for each target appliance to infer its power demand with promising results. DNN overcomes the problems of traditional methods due to its strong abstract feature extraction ability. Moreover, [10] proves that its DNN-based methods are more accurate than the methods based on Combinatorial Optimization (CO) or Factorial Hidden Markov Model (FHMM) through comparison tests.

However, the current DNN-based methods have the following problems: (1) They feed one window of aggregate power sequence into the network each time, which shares the same time frame with the output. This window is not always able to cover a complete working cycle of the target appliance, which may lead to losing significant feature. (2) A DNN has many weight parameters, thus requires large amounts of training data. A direct way of acquiring training data is sub-metering the target appliance to label real aggregate data. But in practice, it is hard to obtain sufficient data in a short period (e.g. a month). [10] proposes a helpful fully random synthetic method (FRSM) to generate substantial amounts of training data while it may not represent the real power consuming scenarios well.

The method proposed in this paper is aimed to solve these problems and thus improve the accuracy of the DNN-based method. (1) An extended input window (EIW) is proposed to guarantee the input window covering a complete working cycle of the target appliance to avoid feature loss. (2) A background-based synthetic method (BSM) is presented to generate abundant and desirable training data. Furthermore, the extended input neural network method using background-based data generation is established for estimating the electricity consumed by the target appliance.

The improvements on the estimation accuracy by both EIW and BSM are validated through comparison experiments. Compared with [10], the proposed estimating method is more accurate.

2. Deep neural network

The deep neural network in this paper is constructed layer by layer with the following layers. Each layer's output will be received by the next layer as input.

2.1. Fully connected layer

A fully connected (FC) layer receives an input vector $\mathbf{X}=(x_i)_{i=1,\cdots,N_1}\in\mathbb{R}^{N_1}$. Suppose that the layer has N_2 units. Each unit $k\in\{1,\cdots,N_2\}$ is a weight vector $\mathbf{W}^k=(w_i^k)_{i=1,\cdots,N_1}\in\mathbb{R}^{N_1}$ and bias $b^k\in\mathbb{R}$, and $z^k\in\mathbb{R}$ is calculated by formula (1). For $\forall k\in\{1,\cdots,N_2\}$, each z^k goes through the activation function $G(\bullet)$ (Sigmoid or ReLU) to compose the output vector $\mathbf{H}=(h_k)_{k=1,\cdots,N_2}\in\mathbb{R}^{N_2}$ by (2).

$$z^k=\sum_{i=1}^{N_1}w_i^k x_i+b^k \quad \in\mathbb{R},\forall k\in\{1,\cdots,N_2\} \tag{1}$$

$$h_k=G(z^k) \quad \in\mathbb{R},\forall k\in\{1,\cdots,N_2\} \tag{2}$$

2.2. 1D convolution layer

Different from FC layers, a 1D convolution layer (Conv1D) [12] receives an L_1-length input sequence with N_1 channels, which is a matrix $\mathbf{X}=(x_{i,j})_{i=1,\cdots,L_1}^{j=1,\cdots,N_1}\in\mathbb{R}^{L_1\times N_1}$. The layer has N_2 filters of length l and depth N_1 (same to the number of input channels). Each filter $k\in\{1,\cdots,N_2\}$ is a weight matrix $\mathbf{W}^k=(w_{i,j}^k)_{i=1,\cdots,l}^{j=1,\cdots,N_1}\in\mathbb{R}^{l\times N_1}$ and bias $b^k\in\mathbb{R}$, and moves across the length of the input sequence by stride s as shown in Fig.1. At each position $i\in\{1,\cdots,L_2\}$ (L_2 denotes the number of positions), $z_i^k\in\mathbb{R}$ is calculated with $\mathbf{W}^k$, b^k, and row $(i-1)\times s+1$ to $(i-1)\times s+l$ in $\mathbf{X}$ by formula (3). j and j' denote the row and column of $\mathbf{W}^k$. For $\forall i\in\{1,\cdots,L_2\}$ and $\forall k\in\{1,\cdots,N_2\}$, each z_i^k goes through $G(\bullet)$ (ReLU) to compose the output matrix $\mathbf{H}=(h_{i,k})_{i=1,\cdots,L_2}^{k=1,\cdots,N_2}\in\mathbb{R}^{L_2\times N_2}$ by (4).

$$z_i^k = \sum_{j'=1}^{N_1}\sum_{j=1}^{l} w_{j,j'}^k x_{(i-1)\times s+j,j'} + b^k \quad \in \mathbb{R}, \forall i \in \{1,\cdots,L_2\}, \forall k \in \{1,\cdots,N_2\} \tag{3}$$

$$h_{i,k} = G(z_i^k) \quad \in \mathbb{R}, \forall i \in \{1,\cdots,L_2\}, \forall k \in \{1,\cdots,N_2\} \tag{4}$$

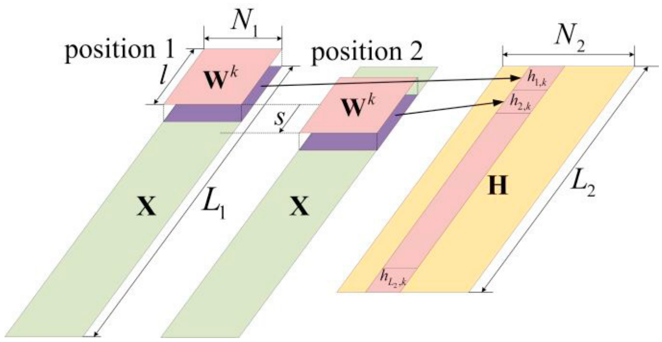

Fig. 1. The calculation of a 1D convolution layer

A 1D max pooling layer (Maxpool1D) can reduce the number of parameters and computation in the network. Suppose that it has a filter of size *l*, and receives an L_1-length input sequence with N channels. The filter moves across the length of the sequence by stride *s*. At each position $i \in \{1,\cdots,L_2\}$ and for each channel $j \in \{1,\cdots,N\}$, $h_{i,j} \in \mathbb{R}$ is obtained by formula (5) to compose matrix $\mathbf{H} = (h_{i,j})_{i=1,\cdots,L_2}^{j=1,\cdots,N} \in \mathbb{R}^{L_2 \times N}$, which is the output sequence.

$$h_{i,j} = max(\{x_{(i-1)\times s+k,j} | \forall k \in \{1,\cdots,l\}\}) \quad \in \mathbb{R}, \forall i \in \{1,\cdots,L_2\}, \forall j \in \{1,\cdots,N\} \tag{5}$$

2.3. Forward Propagation and Backward Propagation

Forward propagation calculates the output of the neural network after it receives an input. Starting at the first layer, each layer's output is calculated by formulas (1)-(5) and passed to the next until the output of the last layer is obtained.

Backward propagation [12,13] is intended to train the network. In each training iteration, a training sample is fed into the neural network. The output of the network and loss function are calculated by forward propagation. Then, a backward propagation calculates the gradients and updates weights and biases to minimize the loss function.

3. The consumption estimating method

3.1. Extended input window neural network

Appliances usually have a specific working cycle which is distinct from others. For example, a washing machine has a specific working process (washing-rest-rinse-spin), and the associated power consumption is shown in Fig. 2. Current studies feed one window (longer than one working cycle duration) into the network, which shares the same time frame with the output as shown in Fig. 2 (a). When the window is at the red position, the network sees a complete working cycle. However, the window may not be placed at this ideal position in practice. If the window is at the shallow red position, the network can only see a partial cycle, which may lead to losing significant feature.

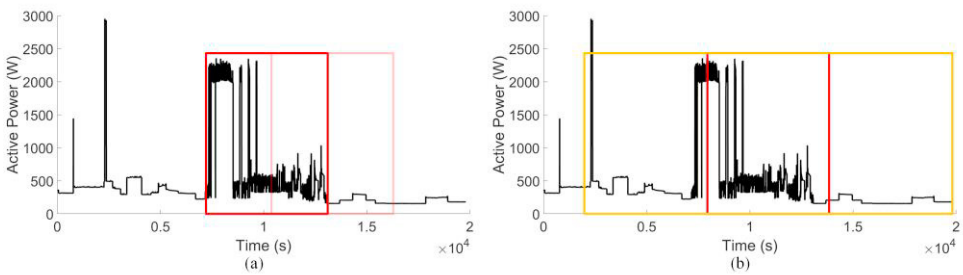

Fig. 2. The input windows comparison

Therefore, the EIW method is proposed, extending the original red window (core window) to the yellow one to supplement the remaining parts of a complete working cycle missed by the original red window, as shown in Fig. 2 (b). The extended length in both sides is the same to the core window length, which is longer than one working cycle (e.g. 1 hour for washing machine, and 20 minutes for kettle). The network receives the whole yellow window of aggregate data and makes estimation for the target appliance's electricity consumption in only the red core window. This ensures the network always sees a complete working cycle no matter which part of it is included by the core window, making full use of the feature and hence improving the estimation accuracy. The EIW neural network for estimating electricity consumption is shown in Fig. 3. Each neural network is for only one target appliance.

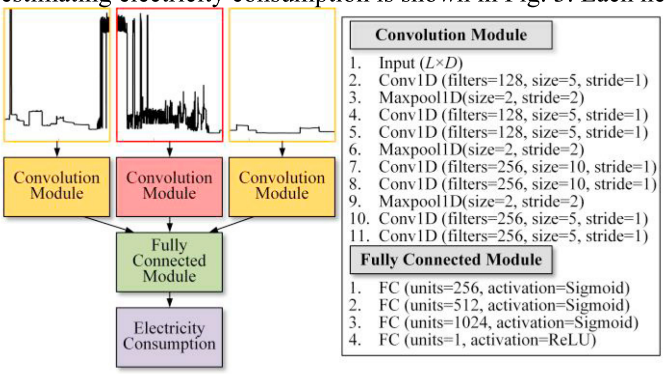

Fig. 3. Extended input window neural network architecture

Algorithm

Cut out a 24-hour-long sequence $\mathbf{P}_{syn}$ from $\mathbf{P}_{back}$
$i = 1$
while = $i \leq$ length($\mathbf{P}_{syn}$)
  Generate a random number q that ranges from 0 to 1
  if $q < p$
    Select an operation curve $\mathbf{P}_{curve}$ randomly
    for j = 1:length($\mathbf{P}_{curve}$)
      $\mathbf{P}_{syn}[i] = \mathbf{P}_{syn}[i] + \mathbf{P}_{curve}[j]$
      $i = i + 1$
    end
  else
    $i = i + 1$
  end
end

Fig. 4. Synthetic method algorithm

In the architecture above, L is the length of the input power sequence, and D is the number of its channels (equals 1 since only active power is used in this paper). In other words, the input sequence corresponds to an L-length aggregate active power sequence. Power refers to active power in latter description. The last FC layer's output is the estimate for the target appliance's electricity consumption in the core window.

The whole extended window is divided into three sub-windows, which go through three different convolution modules, aimed at helping the network distinguish the core window from the extended parts. Experiment evaluation has shown that feeding an entire EIW into one convolution module does not perform as well as using three convolution modules. It may be caused by the fact that the network cannot accurately estimate the electricity consumption by clearly allocating the part of the operation cycle in the core window.

3.2. Background-based data generation

For training the deep neural network and controlling overfitting, large amounts of labeled training data need to be prepared. The real aggregate data labeled by sub-metering the target appliance is usually not sufficient enough. Even if the target appliance works once a day, there will be only 30 instances of its usage in a month. [10] proposes the FRSM, which adds various appliances' operation curves (the power drawn by a single appliance over its one complete operation cycle) in a fully random way to generate synthetic aggregate data. It can provide large amounts of training samples. However, this synthetic aggregate data may not represent the real power consuming scenarios in the house well, leading to a decrease of estimation accuracy on real aggregate data.

This paper treats all the non-target appliances' power consumption as the power use background and uses BSM to generate synthetic data. The following BSM procedure is for only one target appliance in each execution. Firstly, an aggregate power sequence $\mathbf{P}_{nilm}$ and the target appliance's power sequence $\mathbf{P}_{ilm}$ in the same period (e.g. a month) are acquired by metering. Secondly, the target appliance's operation curves are extracted from $\mathbf{P}_{ilm}$. And the power use background power sequence $\mathbf{P}_{back}$ is obtained by subtracting $\mathbf{P}_{ilm}$ from $\mathbf{P}_{nilm}$. Thirdly, a 24-hour-long synthetic aggregate power sequence $\mathbf{P}_{syn}$ is generated by the *Algorithm* shown in Fig. 4. Probability p is set to keep a balance between the number of zero and non-zero electricity consumption samples of the target appliance. Finally, $\mathbf{P}_{syn}$ is split into windows, and each window is a training sample. Repeating the process can obtain sufficient training data.

The process of BSM is simulating the usage of the target appliance at any time in a period based on the real power consuming scenarios of the house. The data generated with BSM obviously has many more instances associated with the target appliance's working state with other appliances than within the real aggregate data. Moreover, the generated data is closer to real situation than the FRSM data due to the fact that it is synthesized based on the measured real power use background, which leads to better performance of the trained neural network.

4. Case study

4.1. Comparison experiment

UK-DALE [14] is used as the dataset for evaluation. Each intrusive sensor installed on an individual appliance samples the active power once every 6 seconds. The non-intrusive sensor samples the aggregate active power once every 1 second and is downsampled to 1/6 Hz for the purpose of this evaluation. 8 major appliances in House1 and House2 are selected as the target appliances for tests. L, the window length, for the washing machines and the dishwashers is set to 600 samples (1 hour), while 200 (20 minutes) for other appliances.

For validating EIW, the network with extended input window ("EIW") is compared with the network using original input window ("OIW"), whose architecture is similar to the one in Fig. 3 but without the yellow windows. For validating BSM, real aggregate data ("Real"), the BSM training data ("BSM"), and the FRSM training data ("FRSM") are used to train the EIW network, respectively. For validating the whole estimation method ("BSM+EIW"), comparison is also made against the Rectangles method ("Rec"), of which the overall performance is the best in [10].

In each test, 4 weeks of data are used as the real aggregate data or the source of background data. Another 4 weeks of data are divided into N core windows with no overlap, and are fed into the trained network for the estimation test.

4.2. Results and discussion

The metrics are given by (6)-(8). y_t denotes actual kWh consumed by the target appliance in core window t, and $\hat{y}_t$ denotes the estimated kWh. *Mean Absolute Error* (MAE, the lower the better), *Accuracy* (Acc) and *Intersection over Union* (IoU, the higher the better) are used as the metrics for evaluation. The results are shown in Table 1.

$$\mathrm{MAE} = \frac{1}{N}\sum_{t=1}^{N}\left|y_t - \hat{y}_t\right| \tag{6}$$

$$\mathrm{Acc} = 1 - \frac{1}{N}\sum_{t=1}^{N}\frac{\left|y_t - \hat{y}_t\right|}{max(y_t, \hat{y}_t)} \tag{7}$$

$$\mathrm{IoU} = \frac{\sum_{t=1}^{N} min(y_t, \hat{y}_t)}{\sum_{t=1}^{N} max(y_t, \hat{y}_t)} \tag{8}$$

Table 1. Experimental results.

Appliance	Method	MAE(kWh)	Acc	IoU	Appliance	Method	MAE(kWh)	Acc	IoU
House1	**BSM+EIW**	**0.0103**	**94.4%**	**73.6%**	House2	**BSM+EIW**	**0.0025**	**98.6%**	**73.6%**
Washing	BSM+OIW	0.0118	91.2%	67.9%	Washing	BSM+OIW	0.0033	98.1%	68.9%
Machine	Real+EIW	0.0171	93.0%	63.4%	Machine	Real+EIW	0.0038	98.3%	61.1%
	FRSM+EIW	0.0107	93.5%	73.5%		FRSM+EIW	0.0048	97.8%	57.4%
	Rec	0.0161	85.1%	59.7%		Rec	0.0068	82.4%	39.7%
House1	**BSM+EIW**	**0.0023**	**99.4%**	**92.4%**	House2	**BSM+EIW**	**0.0024**	**99.3%**	**93.6%**
Dish	BSM+OIW	0.0054	97.5%	77.4%	Dish	BSM+OIW	0.0032	98.7%	91.3%
Washer	Real+EIW	0.0034	98.7%	83.8%	Washer	Real+EIW	0.0048	98.3%	87.8%
	FRSM+EIW	0.0039	98.6%	81.6%		FRSM+EIW	0.0032	99.3%	91.5%
	Rec	0.0099	88.7%	58.5%		Rec	0.0306	84.9%	63.5%
House1	**BSM+EIW**	**0.0006**	**98.7%**	**87.4%**	House2	**BSM+EIW**	**0.0004**	**99.4%**	**94.1%**
Kettle	BSM+OIW	0.0021	96.6%	65.4%	Kettle	BSM+OIW	0.0005	98.9%	90.8%
	Real+EIW	0.0016	95.8%	72.3%		Real+EIW	0.0004	99.4%	94.3%
	FRSM+EIW	0.0009	98.2%	80.9%		FRSM+EIW	0.0005	99.1%	92.4%
	Rec	0.0019	94.3%	62.6%		Rec	0.0012	97.0%	81.8%
House1	**BSM+EIW**	**0.0007**	**98.3%**	**72.7%**	House2	**BSM+EIW**	**0.0005**	**98.2%**	**74.2%**
Toaster	BSM+OIW	0.0012	97.7%	54.5%	Microwave	BSM+OIW	0.0010	96.8%	57.6%
	Real+EIW	0.0016	95.2%	48.8%		Real+EIW	0.0006	98.1%	73.8%
	FRSM+EIW	0.0009	98.3%	68.3%		FRSM+EIW	0.0008	96.5%	69.8%
	Rec	0.0021	89.2%	40.0%		Rec	0.0014	92.4%	48.1%

According to the comparison between "BSM+EIW" and "BSM+OIW", the neural network that adopts EIW performs better than the other. Washing machines and dish washers have strong relevance between different parts of their one working cycle. EIW can ensure the full use of this relevance, avoiding missing or inaccuracy caused by the network seeing only partial working cycle. And for appliances that have simple operation curves (e.g. kettle), EIW can help network identify their working duration in the core window to improve the estimation accuracy.

Comparison between "BSM+EIW" and "Real+EIW" indicates that IoU of the former is at least 10% higher on House1 and is 12% higher on the dishwasher of House2. Comparison between "BSM+EIW" and "FRSM+EIW" indicates that the former is better than the latter overall.

The results show that both EIW and BSM have improvements on the estimation accuracy. In addition, according to the comparison against "Rec", "BSM+EIW", the proposed estimating method, produces more accurate estimates for the electricity consumption of the target appliance than the Rectangles. Therefore, the proposed extended input neural network estimating method using background-based data generation is validated to be effective.

5. Conclusion

Non-intrusive load monitoring can make estimation for the electricity consumption of each individual appliance, which is of great significance for power grid operation and user-side energy conservation. Deep neural network is an efficient method for NILM. However, there are problems on utilization of working feature and acquisition of sufficient training data in current studies. This paper establishes an extended input neural network method using background-based data generation for estimating electricity consumption. On the one hand, it avoids feature loss. On the other hand, it increases training data abundance. The experimental results indicate that the accuracy is improved by solving these problems and the proposed estimation method is superior to other DNN-based methods.

Acknowledgements

This work was supported by the China National Key Research & Development Project (No. 2018YFB0904502)

References

[1] G. W. Hart. Nonintrusive appliance load monitoring. *Proc. IEEE*, vol. 80, no. 12, pp. 1870-1891, Dec. 1992.

[2] S. Darby. The effectiveness of feedback on energy consumption. A review for Defra of the literature on metering, billing and direct displays. Environmental Change Inst., Univ. Oxford, Oxford, U.K., Tech. Rep., 2006.

[3] S. B. Leeb, S. R. Shaw, J. L. Kirtley, Jr. Transient Event Detection in Spectral Envelope Estimates for Nonintrusive Load Monitoring. *IEEE Trans. Power Delivery*, p. 1200, Jul. 1995.

[4] H. H. Chang. Non-intrusive demand monitoring and load identification for energy management systems based on transient feature analyses. *Energies*, vol. 5, no. 11, pp. 4569-4589, Nov. 2012.

[5] N. Amirach, B. Xerri, B. Borloz and C. Jauffret. A new approach for event detection and feature extraction for NILM. *2014 21st IEEE ICECS.*, Marseille, 2014, pp. 287-290.

[6] B. Liu, W. Luan, Y. Yu. Dynamic time warping based non-intrusive load transient identification. *Applied Energy*, pp. 634-645, June. 2017.

[7] K. Suzuki, S. Inagaki, T. Suzuki, H. Nakamura, and K. Ito. Nonintrusive appliance load monitoring based on integer programming. in *Proc. SICE Annu. Conf.*, Aug. 2008, pp. 2742–2747.

[8] D. Piga, A. Cominola, M. Giuliani, A. Castelletti and A. E. Rizzoli. Sparse Optimization for Automated Energy End Use Disaggregation. *IEEE Trans. Control Systems Technology*, vol. 24, no. 3, pp. 1044-1051, May 2016.

[9] S. Makonin, F. Popowich, I. V. Bajic, B. Gill, and L. Bartram. Exploiting HMM sparsity to perform online real-time nonintrusive load monitoring. *IEEE Trans. Smart Grid*, vol. 7, no. 6, pp. 2575-2585, Nov. 2016.

[10] J. Kelly and W. Knottenbelt. Neural NILM: Deep neural networks applied to energy disaggregation. in *Proc. 2nd ACM Int. Conf. Embedded Syst. Energy-Efficient Built Environ.*, 2015, pp. 55–64.

[11] L. Mauch and B. Yang. A new approach for supervised power disaggregation by using a deep recurrent LSTM network. in *2015 IEEE GlobalSIP*, Orlando, FL, 2015, pp. 63-67.

[12] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner. Gradient-based learning applied to document recognition. in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998.

[13] D. Rumelhart, G. Hinton and R. Williams. Learning Representations by Back-Propagating Errors. *Nature*, vol. 323, pp. 533-536, Oct. 1986.

[14] 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, Mar. 2015.