A hierarchical HMM framework for home appliance modelling

Weicong Kong

March 1, 2020

Menu

Abstract

why

- Proper individual home appliance modeling is critical to the performance of NILM.
- This model aims to provide better representation for those appliances that have multiple built-in modes.

what

- The dynamic Bayesian network representation of such an appliance model is built.
- A forward-backward algorithm, which is based on the framework of expectation maximization, is formalized for the HHMM fitting process.

how

- Tests on publically available data show that the HHMM and proposed algorithm can effectively handle the modeling of appliances with multiple functional modes, as well as better representing a general type of appliances.
- A disaggregation test also demonstrates that the fitted HHMM can be easily applied to a general inference solver(particle filter) to outperform conventional hidden Markov model in the estimation of energy disaggregation.

NILM problems during steps

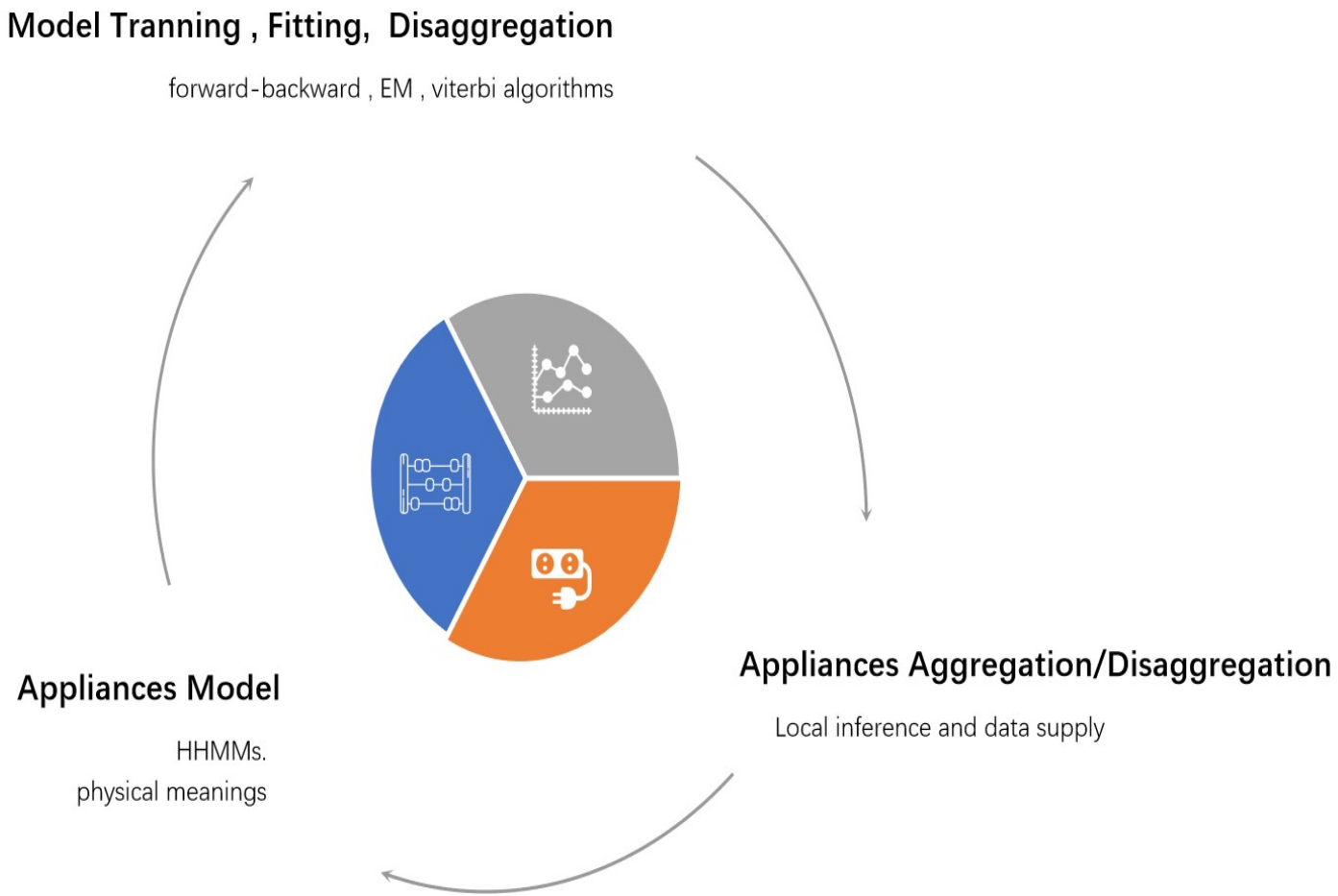

Figure 1: NILM problems during steps

- Problem statement $y_t = \sum_k f_k(x_t^k) + e_t$

Introduction

HMM with low frequency data(first 3 paragraphs)

- The advent of smart meter infrastructure has provided plentiful information of studying load characteristics, but research in mining such big data to provide actionable insights is still at the early stage.
- Load disaggregation is one of the key directions...
- low frequency sampling data with HMM model is attracting attention.

HMM not ok with two/multiple mode 4th paragraph

- but ... have difficulty to distinguish the standby state and the off state.
- for multiple mode appliances,...not be captured, depending on user's behaviour
- re-usability of such model is compromised in the load disaggregation problem

HMM variant with hierarchical structure

- in the area of speech, Cocktail Party Problem
- hierarchical HMM is adopted
- two Markov chain, upper chain(the mode of appliance), lower chain(power profile of the specific mode)

previous HHMM

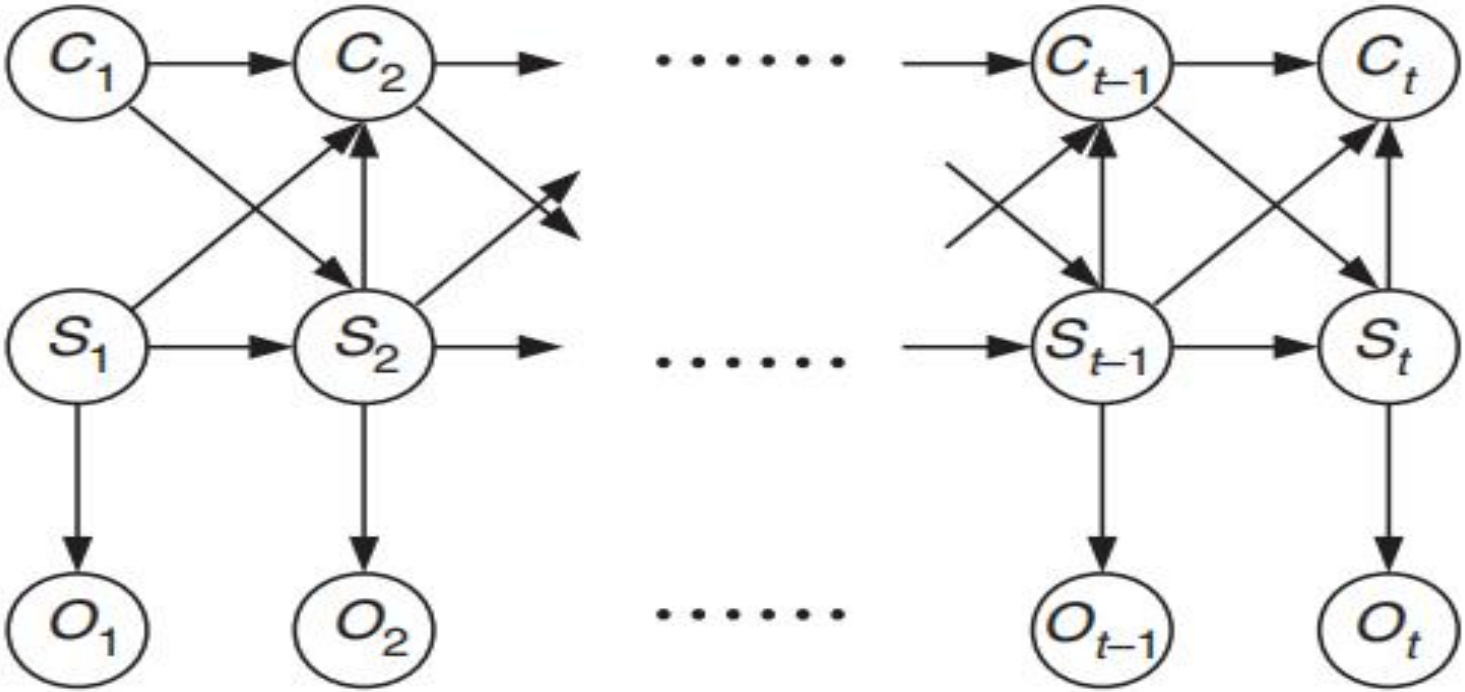

Figure 2: prevois HHMM,model as DBN

- Wong, Yung Fei, Thomas Drummond, and Y. A. Şekercioğlu. ”Real-time load disaggregation algorithm using particle-based distribution truncation with state occupancy model.” Electronics Letters 50.9 (2014): 697-699.

previous Sequential Monte Carlo/particle filter for localization simulation

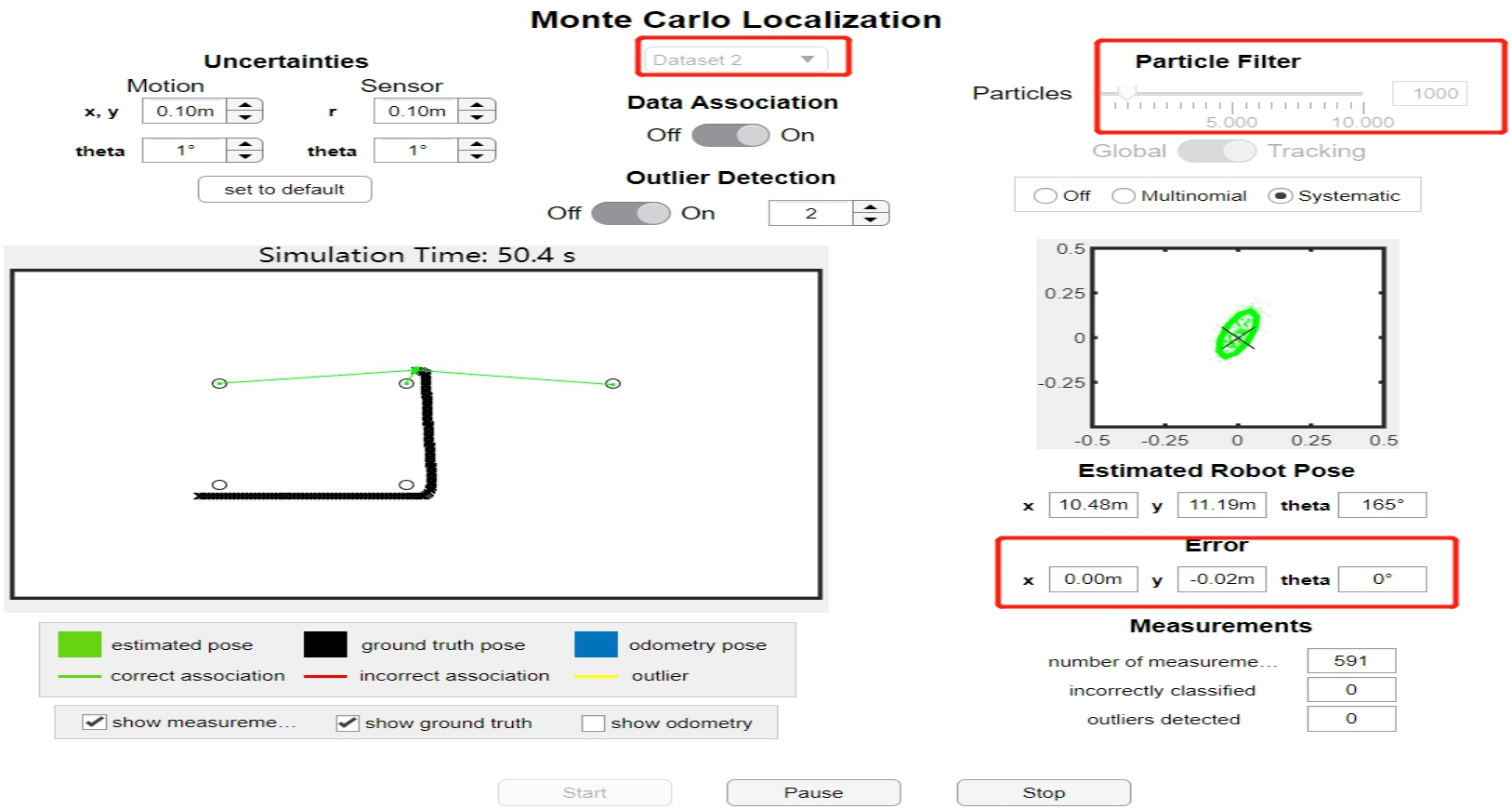

Figure 3: previous Monte Carlo/particle filter for localization simulation

Monte Carlo Localization flow

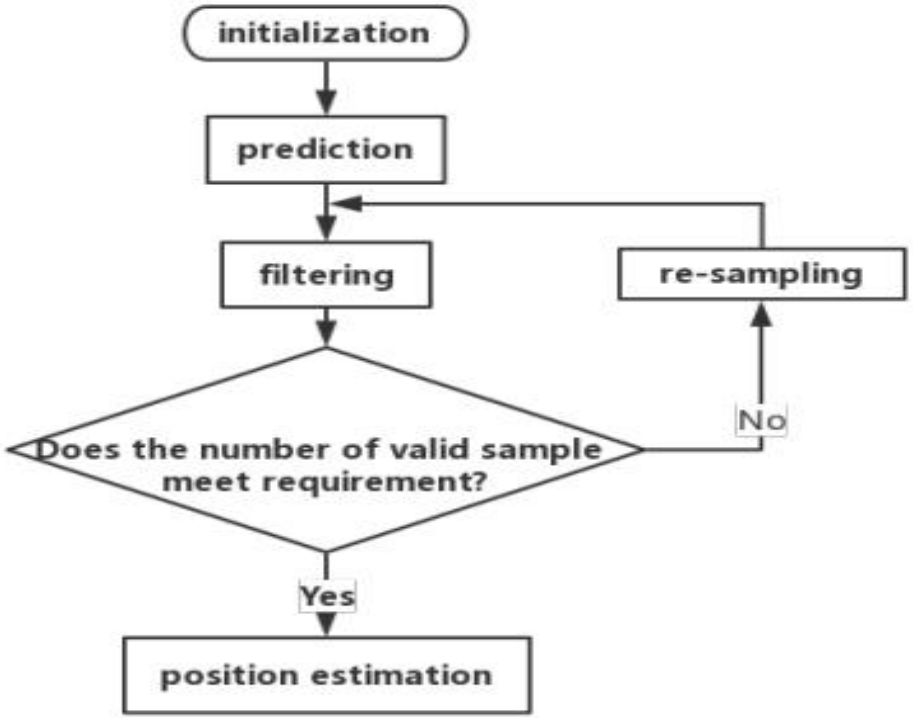

Figure 4: Monte Carlo Localization flow

Why The particle filter solver for NILM with DBN model

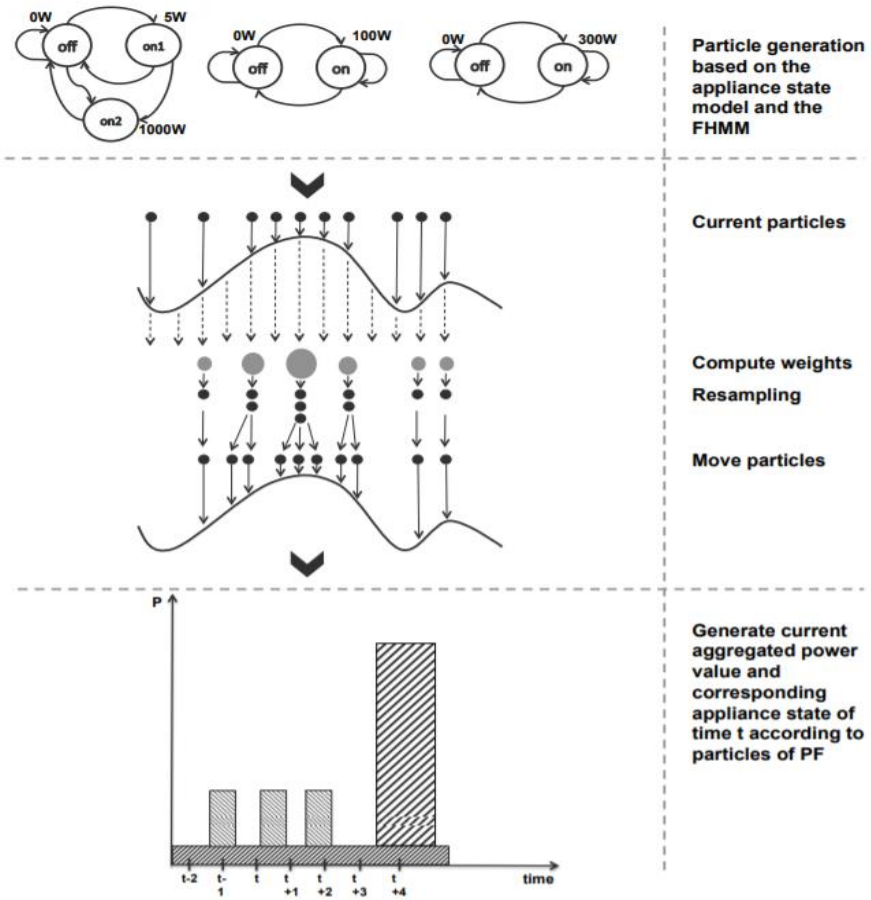

Figure 5: NILM using particle filter

- non-linear, non-Gaussian Ditribution

Hidden Markov Model

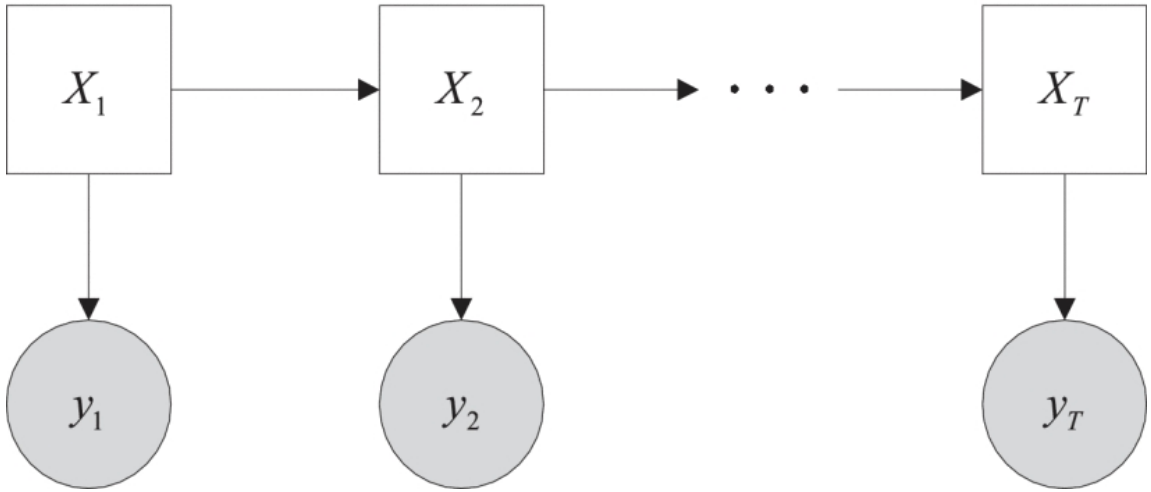

Figure 6: Graphical illustration of a HMM.

Hmm provides a good framework to describe the dynamic through discretised time series of individual appliances.

$$\pi_i = P(X_1 = i) \tag{1}$$

$$A_{i,j} = P\left(X_t = j \middle| X_{t-1} = i\right) \tag{2}$$

$$P\left(y_t \middle| X_t = i\right) \sim \mathcal{N}(\mu_i,\ \sigma_i) \tag{3}$$

Hierarchical Hidden Markov Model(HHMM)

- an extension of HMM to hierarchical hidden markov model
- dynamic bayesian network

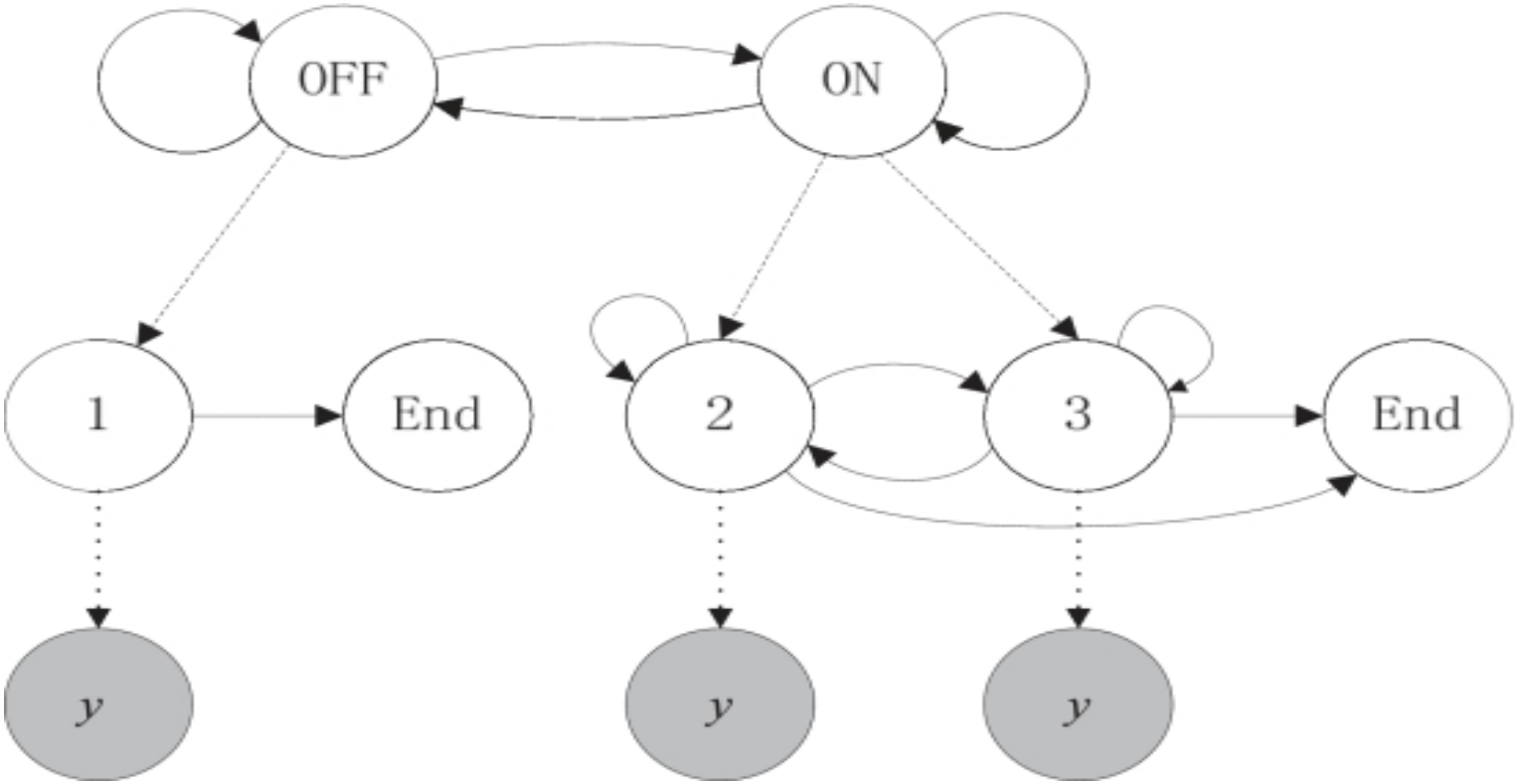

Figure 7: The state transition diagram for a two-level HHMM, modelling a 3-state appliance.

HHMM

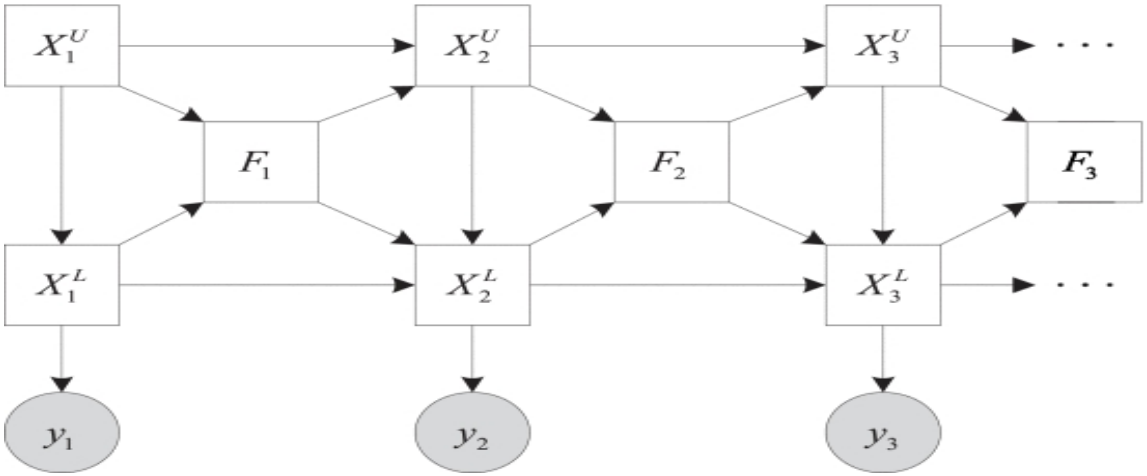

Figure 8: Graphical illustration of a two-level HHMM.

$$P\left(X_t^L = j \,\middle|\, X_{t-1}^L = i\ , F_{t-1} = f, X_t^U = \kappa\right) = \begin{cases} \tilde{A}_\kappa^L(i,j) & \text{if } f = 0 \\ \pi_\kappa^L(j) & \text{if } f = 1 \end{cases} \tag{4}$$

1. X_t^U , X_{t-1}^L: state variables of upper/lower level at time t
2. F_{t-1}: binary auxiliary variable, 1: lower level HMM finished,0: not finished
3. $\tilde{A}_\kappa^L(i,j)$ is rescaled version of $A_\kappa^L(i,j)$
4. $\pi_\kappa^L(j)$ is the distribution for initial state for the low level

HHMM eq5-9

1. The relationship between the subHMM transition matrix and rescaled matrix:

$$\tilde{A}_{\kappa}^{L}(i,j)\left(1-A_{\kappa}^{L}(i,end)\right)=A_{\kappa}^{L}(i,j) \tag{5}$$

$A_{\kappa}^{L}(i,end)$ is the probability of a state being terminiated from state i

$\sum_{j=1}^{K}\tilde{A}_{\kappa}^{L}(i,j)=1$

$$P\left(F_t=1\middle|X_t^L=i,X_t^U=\kappa\right)=A_{\kappa}^{L}(i,end) \tag{6}$$

$$P\left(F_t=0\middle|X_t^L=i,X_t^U=\kappa\right)=1-A_{\kappa}^{L}(i,end) \tag{7}$$

$$P\left(X_t^U=\kappa\middle|X_{t-1}^U=\eta,F_{t-1}=f\right)=\begin{cases}\delta(\eta,\kappa) & \text{if } f=0\\ A^U(\eta,\kappa) & \text{if } f=1\end{cases} \tag{8}$$

$$P\left(X_1^U=\kappa\right)=\pi^U(\kappa) \tag{9}$$

Model fitting

- θ denote the of paramters, $\boldsymbol{\theta}$ is the estimated value, $y = y_{1:T}$ is the whole observed sequence

$$\overline{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} P\left(\boldsymbol{y}|\boldsymbol{\theta}\right) \tag{10}$$

  the famous algorithm is expectation maximization (EM) algorithm
- EM for HMM
- EM for HHMM
- HMM fitting by adopting the dynamic bayesian network representation
- for simplicity, $P(y|\theta) = P(y_{1:T})$

Expectation Maximization for HMM

- the first version of forward variable

$$\alpha_t(j) = P\left(X_t = j, y_{1:t}\right) \tag{11}$$

- The normalized version adopted for forward variable in this paper

$$\begin{aligned}\alpha_t(j) &\triangleq \frac{P\left(X_t = j, y_{1:t}\right)}{P\left(y_{1:t}\right)} = P\left(X_t = j \middle| y_{1:t}\right) \\ &\propto \left(\sum_{i=1}^{K} \alpha_{t-1}(i) A_{i,j}\right) P\left(y_t \middle| X_t = j\right)\end{aligned} \tag{12}$$

- backward variable

$$\beta_t(i) \triangleq P\left(y_{t+1:T} \middle| X_t = i\right) = \sum_{j=1}^{K} A_{i,j} P\left(y_{t+1} \middle| X_{t+1} = j\right) \beta_{t+1}(j) \tag{13}$$

- the base case for both forward and backward variables

$$\alpha_1(j) = P\left(y_1 \middle| X_1 = j\right) \pi_j \tag{14}$$

$$\beta_T(i) = 1 \tag{15}$$

Expectation Maximization

- The probability of each state at t given observation sequence

$$\begin{aligned}\gamma_t(i) &\triangleq P\left(X_t = i \middle| y_{1:T}\right) \\ &= \frac{1}{P\left(y_{1:T}\right)} P\left(y_{t+1:T} \middle| X_t = i\right) P\left(X_t = i \middle| y_{1:t}\right) \\ &\propto \alpha_t(i)\beta_t(i) \end{aligned} \tag{16}$$

-

$$\begin{aligned}\xi_t(i,j) &\triangleq P\left(X_{t-1} = i, X_t = j \middle| y_{1:T}\right) \\ &\propto P\left(X_t = j \middle| X_{t-1} = i\right) \times P\left(X_{t-1} = i \middle| y_{1:t-1}\right) \\ &\quad \times P\left(y_{t+1:T} \middle| X_t = j\right) \times P\left(y_t \middle| X_t = j\right) \\ &= A_{i,j}\alpha_{t-1}(i)\beta_t(j)P\left(y_t \middle| X_t = j\right) \end{aligned} \tag{17}$$

- $\gamma_t(i)$ and $\xi_t(i,j)$ is the estimation step of EM

Expectation Maximization

- The maximization step of EM

$$\overline{\pi}_i = \gamma_1(i) \tag{18}$$

$$\overline{A}_{i,j} = \frac{1}{Z} \sum_{t=2}^{T} \xi_t(i,j) \tag{19}$$

$$\overline{y}_t(i) = \gamma_t(i) \times y_t. \tag{20}$$

- parameters by fitting a Gaussian distribution using a maximum likelihood estimate
- Z is a normalizing constant to ensure the transition matrix is a stochastic matrix

Number Underflow

- as the length of sequence grows, the joint probability keeps decreasing.
- the value exceed the maximum length which the computer can represent, resulting the value to be considered to 0
- to deal with this issue, using log
- normalization is also performed in log space to mitigate the effect,

$$\sum_{j} exp(\alpha_t(j)) = 1$$

The log likelihood of model parameters

- the forward variables be rewritten into

$$\begin{aligned}\alpha_t(j) &= P\left(X_t = j \middle| y_{1:t}\right) \\ &= \frac{1}{P\left(y_t \middle| y_{1\,:\,t-1}\right)} P\left(X_t = j, y_t \middle| y_{1\,:\,t-1}\right)\end{aligned} \tag{21}$$

- letting $c_t = P\left(y_t | y_{1\,:\,t-1}\right)$,then

$$c_t = \sum_j P\left(X_t = j, y_t\big|_{1\,:\,t-1}\right) \tag{22}$$

$$P\left(X_t = j, y_t \middle| y_{1:t-1}\right) = \left(\sum_{i=1}^{K} \alpha_{t-1}(i) A_{i,j}\right) P\left(y_t \middle| X_t = j\right)$$

- likelihood of observation can be calculated

$$\begin{aligned}P\left(y_{1:T}\right) &= P\left(y_1\right) P\left(y_2 \middle| y_1\right) \cdots P\left(y_T \middle| y_{1:T-1}\right) \\ &= \prod_{t=1}^{T} c_t\end{aligned} \tag{23}$$

Expectation Maximization for HHMM

- the set of parameters for a HHMM

$$\boldsymbol{\Theta} = \left\{ A^U, \pi^U, \left\{ \tilde{A}_\kappa^L \right\}, \left\{ \pi_\kappa^L \right\}, \left\{ A_\kappa^L \left(:, end\right) \right\}, \boldsymbol{O} \right\} \tag{24}$$

  every level has its unique state labels
- upper level ON,OFF
- lower level 1,2,3
- the vertical transition matrix is sparse because the lower level is 1 where the upper level is off
- i.e.$P(X_t^L = 1 | X_t^U = OFF) = 1$
- $P(X_t^L = 2 | X_t^U = OFF) = 0$, $P(X_t^L = 3 | X_t^U = OFF) = 0$

The forward variables of EM in HHMM

- According to this stable labelling setting

$$\begin{aligned}\alpha_t\left(j,\kappa,f\right) &\triangleq P\left(X_t^L=j,X_t^U=\kappa,F_{t-1}=f\middle|y_{1:t}\right)\\ &\propto P\left(y_t\middle|X_t^L=j\right)\sum_i\sum_\eta h\left(i,\eta\right)\sum_g\alpha_{t-1}\left(i,\eta,g\right)\end{aligned} \tag{25}$$

- κ is the state of the upper level
- h is an auxiliary function

$$h\left(i,\eta\right)=\begin{cases}\left(1-A_\eta^L\left(i,end\right)\right)\delta\left(\eta,\kappa\right)\tilde{A}_\kappa^L\left(i,j\right) & \text{if } f=0\\ A_\eta^L\left(i,end\right)A^U\left(\eta,\kappa\right)\pi_\kappa^L\left(j\right) & \text{if } f=1\end{cases} \tag{26}$$

- the initial forward variable

$$\alpha_1\left(j,\kappa,f\right)=\begin{cases}0 & \text{if } f=0\\ \pi^U\left(\kappa\right)\pi_\kappa^L\left(j\right)P\left(y_1\middle|X_1^L=j\right) & \text{if } f=1\end{cases} \tag{27}$$

The backward variables

- backward variable

$$\begin{aligned}\beta_t(i,\eta,f) &\triangleq P\left(y_{t+1:T}\middle|X_t^L=i,X_t^U=\eta,F_{t-1}=f\right)\\ &=\sum_j P\left(y_{t+1}\middle|X_{t+1}^L=j\right)\sum_\kappa\left(\left(1-A_\eta^L(i,end)\right)\right.\\ &\quad\times\delta(\eta,\kappa)\times\tilde{A}_\kappa^L(i,j)\times\beta_{t+1}(j,\kappa,0)+A_\eta^L(i,end)\\ &\quad\left.\times A^U(\eta,\kappa)\times\pi_\kappa^L(j)\times\beta_{t+1}(j,\kappa,1)\right)\end{aligned}\tag{28}$$

- the initial backward variable

$$\beta_T(i,\eta,f)=1\tag{29}$$

Sufficient Statistics 1

- horizontal transition probability

$$\begin{aligned}\xi_t^h(i,j,\kappa) &= P\left(X_{t-1}^L = i, X_t^L = j, X_t^U = \kappa, F_{t-1} = 0 \middle| y_{1:T}\right) \\ &\propto \sum_{\eta}\sum_{g} \alpha_{t-1}(i,\eta,g) \times \left(1 - A_\eta^L(i,end)\right) \\ &\quad \times \delta(\eta,\kappa) \times \tilde{A}_\kappa^L(i,j) \times P\left(y_t \middle| X_t^L = j\right) \\ &\quad \times \beta_t(j,\kappa,0) \end{aligned} \tag{30}$$

- vertical transition probability

$$\begin{aligned}\xi_t^v(j,\kappa) &= P\left(X_t^L = j, X_t^U = \kappa, F_{t-1} = 1 \middle| y_{1:T}\right) \\ &\propto \alpha_t(j,\kappa,1)\,\beta_t(j,\kappa,1)\end{aligned} \tag{31}$$

- smoothed state probability

$$\begin{aligned}\gamma_t(j,\kappa,f) &= P\left(X_t^L = j, X_t^U = \kappa, F_{t-1} = f \middle| y_{1:T}\right) \\ &\propto \alpha_t(j,\kappa,f)\,\beta_t(j,\kappa,f)\end{aligned} \tag{32}$$

Sufficient Statistics 2

- end transition probability

$$\begin{aligned}\xi_t^e(i,\eta,1) &= P\left(X_t^L = i, X_t^U = \eta, F_t = 1 \middle| y_{1:T}\right) \\ &\propto \sum_j \sum_\kappa \sum_f \alpha_t(i,\eta,f) \times A_\eta^L(i,end) \\ &\quad \times A^U(\eta,\kappa) \times \pi_\kappa^L(j) \\ &\quad \times P\left(y_t \middle| X_t^L = j\right) \times \beta_{t+1}(j,\kappa,1) \end{aligned} \tag{33}$$

- $\xi_t^e(i,\eta,1)$ denotes the probability that subHMM ends
- $\xi_t^e(i,\eta,0)$ denotes the probability that subHMM not ends

$$\begin{aligned}\xi_t^e(i,\eta,0) &= P\left(X_t^L = i, X_t^U = \eta, F_t = 0 \middle| y_{1:T}\right) \\ &\propto \sum_j \sum_\kappa \sum_f \alpha_t(i,\eta,f) \\ &\quad \times \left(1 - A_\eta^L(i,end)\right) \times \delta(\eta,\kappa) \\ &\quad \times \tilde{A}_\kappa^L(i,j) \times P\left(y_{t+1} \middle| X_{t+1}^L = j\right) \\ &\quad \times \beta_{t+1}(j,\kappa,0) \end{aligned} \tag{34}$$

Sufficient Statistics 3

- horizontal transition probability for the upper level

$$\begin{aligned}\chi_t(\eta,\kappa) &= P\left(X_{t-1}^U=\eta, X_t^U=\kappa, F_{t-1}=1\middle|y_{1:T}\right)\\ &\propto \sum_i\sum_j\sum_g \alpha_{t-1}(i,\eta,g)\times A_\eta^L(i,end)\\ &\quad\times A^U(\eta,\kappa)\times \pi_\kappa^L(j)\\ &\quad\times P\left(y_t\middle|X_t^L=j\right)\times\beta_t(j,\kappa,1)\end{aligned} \tag{35}$$

The maximization step

- the contribution of an observation to a concrete state i

$$\overline{y}_t(i) = \left(\sum_{\kappa} \gamma_t(i, \kappa) \right) \times y_t \tag{36}$$

- the lower level transition parameters

$$\overline{\tilde{A}}_{\kappa}^{L}(i, j) = \frac{1}{Z_h} \sum_{t=2}^{T} \xi_t^h(i, j, \kappa) \tag{37}$$

- Z_h is the normalizing constant to ensure the is a stochastic matrix
- the end transition parameters

$$\overline{A}_{\kappa}^{L}(i, end) = \frac{1}{Z_e} \sum_{t=1}^{T-1} \xi_t^e(i, \kappa, 1) \tag{38}$$

- Z_e is the normalizing constant matrix for the end transitions

$$Z_e(i, \eta) = \sum_{t=1}^{T-1} \sum_{g} \xi_t^e(i, \kappa, g) \tag{39}$$

The maximization step

- the initial state distribution parameters at the lower level

$$\pi_{\kappa}^{L}(j) = \frac{1}{Z_v} \sum_{t=1}^{T} \xi_t^v (j, \kappa) \tag{40}$$

- the transition parameter at the upper level

$$A^U (\eta, \kappa) = \frac{1}{Z_U} \sum_{t=2}^{T} \chi_t (\eta, \kappa) \tag{41}$$

The proposed EM algorithm for HHMM

Algorithm I: The EM algorithm for HHMM

Inputs: $\boldsymbol{y}$, K^U, $K^L = \{K_1^L, \ldots, K_\kappa^L\}$
Outputs: $\boldsymbol{\Theta}$
Set $iter = 0$
Initialize $\boldsymbol{\Theta}_{\text{iter}}$
Set $iter = iter + 1$
While $iter < maxIter$ and $\Delta lik < threshold$
  For $t = 1$ to T
    // the E-step
    Compute and store forward and backward variables α_t, β_t using (25), (27), (28), (29)
    Compute and store all sufficient statistics ξ_t^h, ξ_t^v, γ_t, ξ_t^e, χ_t using (30), (31), (32), (33), (34), (35)
    // the M-step
    Update $\boldsymbol{\Theta}$ using (36), (37), (38), (40), (41)
    Update the model log-likelihood $loglik_{iter}$ using (23)
    Set $iter = iter + 1$
  End for
End while
Return $C_1, \ldots, C_{N_c}, \mu, \sigma$

Figure 9: The proposed EM algorithm for HHMM

fitting single mode appliances and its benefit

- modelling real world appliances with our proposed model and fitting algorithm
- in NILM area, most neither state the length of the power profiles nor used a rather lengthy sequence for fitting model
- lengthy sequence not represent good generality
- so..only use one typical duty cycle which eliminating the possibility of including the extra information
- test the proposed model using publically available data introduced by..
- typical duty cycle power sequence
- i.e. typical one of washing machine: normal wash, permanent press and delicate wash. The power consumption pattern differ from each other.
- down-sampled to 1 reading per minute

fitting multiple mode appliances and its benefit

- the proposed algorithm is applied ...yield three HHMMs
- three HHMMs assembled to one HHMM
- assembling rule
  1. without further information
  2. upon finishing any of the modes, should go back to the OFF mode

Synthesis test case design

- the proposed HHMM can be learned via EM
- then apply the learned models in the NILM
- the dataset include typical cycles of 7 major appliance types
- the difficulty of NILM are how mixed at same period
  - randomly mix a random set of 7 appliance into a short 3 hour peroid
  - generate sufficiently complicated test cases
  - in order to present the varying complexity and diversity of reallife situations, 30 test cases

The particle filter solver

- the particle filter solver adopted to NILM with HMM [12][24][25]
- the pros of the PF solver: linear scalability with number of appliances
- the con of the PF solver: slow
- have not PF solver with HHMM, but in this paper
- PF for HHMM should have an upper state sequence sampled in real time. The lower state transition governed by the upper.
- the upper state for appliance n at time t of particle m:

$$x_t^{U(n)}[m] = \begin{cases} x_{t-1}^{U(n)}[m] & \text{if } f = 0 \\ \sim A^{U(n)}_{x_{t-1}^{U(n)}[m],*} & \text{if } f = 1 \end{cases} \tag{42}$$

- $\sim$, the distribution of random variable, m is the particle index, A is the state transition matrix
-

$$x_t^{L(n)}[m] \sim \begin{cases} \tilde{A}^{L(n)}_{x_{t-1}^{U(n)}[m]}\left(x_t^{L(n)}[m],*\right) & \text{if } f = 0 \\ \pi^{L(n)}_{\kappa x_{t-1}^{U(n)}[m]} & \text{if } f = 1 \end{cases} \tag{43}$$

The particle filter solver

- assuming a load disaggregation problem with N appliances, T time steps
- each appliance with K^U upper state and K^L lower state

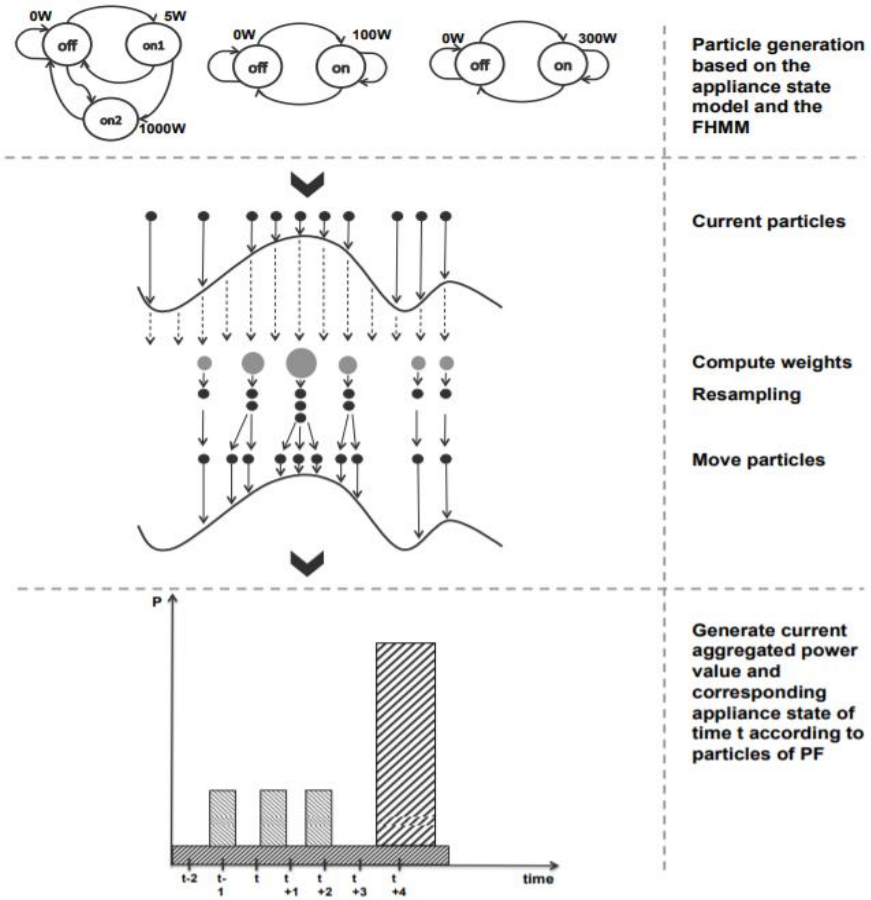

Figure 10: NILM using particle filter

The evaluation metric

- there are numberous evaluation metrics for load disaggregation
- not only the on/off events of each appliance, but also the amount of power consumption
- The metric is calculated by

$$Acc = 1 - \frac{\sum_{t=1}^{T}\sum_{n=1}^{N}\left|\hat{y}_t^{(n)} - y_t^{(n)}\right|}{2\sum_{t=1}^{T} y_t} \tag{44}$$

- $\hat{y}_t^{(n)}$ denotes the estimate of power consumption for the ith appliance at time t
- $y_t^{(n)}$ denotes the actual power consumption for the ith appliance
- y_t denotes the observed power consumption sequence

results and comparison

- 30 cases 4 different scenarios using particle filter solver
- the number of particles is fixed to 5000
- from table v, HHMM outperformance the conventional HMM
- from fig.11, HHMM more stabe than HMM
- fig.12-16, more benefit of HHMM

real world test case REDD dataset

further justify proposed model, apply PF with HHMM to the REDD dataset

1. not provide the detailed information about appliance modes
2. real world dataset confirm the potential of HHMM
3. although the performance decreases as the number of appliances increase
4. one of the causes is that most appliances in real world pratice have muliple modes but modelled as single-mode

Conclusion

- this paper
  - inpired by the speech recognition
  - more concise EM for HHMM fitting
  - comprehensive test verified better result in NILM
- theoretical progress
  - the efficiency of sequential Monte Carlo (SMC) for performing inference in structured probabilistic models
  - and the flexibility of deep neural networks to model complex conditional probability distributions
  - i.e.AUTO-ENCODING SEQUENTIAL MONTE CARLO (ICLR 2018)
- Scenario, inference in embedded system, time consuming before the deadline, in order real time control
  - need the explainable inference model
  - training or fitting model parameter in the cloud
- future work
  - dataset format:
  - model:HHMM extend to combine with other sophisticated i.e. HSMMs, HMMs
  - fitting:more efficient solver with HHMM
  - solver:
  - evaluation:...plenty of space
  - loc-aware disaggregation

Thank You!