Variational BOLT: Approximate Learning in Factorial Hidden Markov Models with Application to Energy Disaggregation

Henning Lange, Mario Bergés
Carnegie Mellon University
Pittsburgh, PA, USA

Abstract

The learning problem for Factorial Hidden Markov Models with discrete and multi-variate latent variables remains a challenge. Inference of the latent variables required for the E-step of Expectation Minimization algorithms is usually computationally intractable. In this paper we propose a variational learning approach mimicking the Baum-Welch algorithm. By approximating the filtering distribution with a variational distribution parameterized by a recurrent neural network, the computational complexity of the learning problem as a function of the number of hidden states can be reduced to quasilinear instead of quadratic time as required by traditional algorithms such as Baum-Welch whilst making minimal independence assumptions. We evaluate the performance of the resulting algorithm, which we call Variational BOLT, in the context of unsupervised end-to-end energy disaggregation. Specifically, we conduct experiments on the publicly available REDD dataset and show competitive results when compared with a supervised inference approach and state-of-the-art results in an unsupervised setting.

Introduction

Because of its potential to discover and unlock energy saving opportunities in buildings, the problem of energy disaggregation (Hart 1992) has received increased interest from various academic communities. The objective of this single-channel source separation problem, also known as non-intrusive load monitoring (NILM), is to infer appliance-level power consumption information given data from only a single sensing point at the main electrical panel of a building. The observed aggregate power measured at the main panel constitutes the sum of the power of the individual appliances. Since, appliance states (e.g., *on*, *off*) can be assumed to evolve independently and the aggregate observation is dependent on the joint state of all appliances, Factorial Hidden Markov Models (FHMM) (Ghahramani and Jordan 1997) have emerged as a prominent model for the generative process of the aggregate observations. However, because latent states become conditionally dependent given the observations, exact inference of the posterior, i.e. the distribution of appliance states given the aggregate observation, is assumed to be intractable. Numerous approximate inference techniques have been proposed and employed in the past to tackle this problem. Authors in (Kolter and Jaakkola 2012) introduced the one-at-a-time constraint that postulates that only a single appliance can change its state at any given time. This constraint allows for posterior inference by either posing the problem as an integer programming problem (Kolter and Jaakkola 2012) or by truncating the Viterbi algorithm (Lange and Bergés 2016b). However, these approaches were focused on the decoding rather than the learning problem. On the other hand, (Jia, Gao, and Spanos 2015) proposed a solution to the learning problem based on MCMC sampling but this approach seems to struggle with slow mixing of the posterior.

The model introduced in this paper makes use of a highly tractable auxiliary distribution that approximates the true filtering distribution. This tractable distribution is parameterized by a deep recurrent neural network, specifically stacked LSTMs (Hochreiter and Schmidhuber 1997). We build on recent results showing how neural networks in conjunction with Variational Inference can be used as a powerful tool for statistical inference (Kingma and Welling 2013). This combination is favorable because Variational Inference allows one to pose statistical inference as a (non-linear) optimization problem and neural networks have become a dominant approach for non-linear optimization.

Because we assume the latent variables to be binary (i.e. a Bernoulli auxiliary distribution) we face the additional difficulty of dealing with non-conjugate discrete distributions, which pose significant challenges in the context of neural networks and variational inference (Kingma and Welling 2013). As we will show later, this problem is circumvented by directly approximating the expectation of the true filtering distribution and introducing a loss function that directly penalizes the neural network outputs for deviating from that approximation. By making use of a tractable auxiliary distribution and making minimal independence assumptions (i.e. only posterior independence of the latent variables at a single time point) computations required for the filtering distribution that are typically quadratic in the number of hidden states can be reduced to quasilinear time. This ultimately allows modeling rich temporal dependencies between latent variables whilst keeping computational costs low. Although we focus on the application of energy disaggregation in this paper, our proposed method could find applications in other fields where FHMMs with binary latent states are employed such as in certain Bioinformatics problems, e.g., (Stanculescu, Williams, and Freer 2014; Asif and Sanguinetti 2011).

The decreased computational costs for inference afford our solution with the additional advantage of addressing security and privacy concerns associated with energy disaggregation solutions. In other words, disaggregation can, in principle, be carried out in real-time on cheap off-the-shelf embedded hardware located within the premises.

In the next section, we introduce Factorial Hidden Markov Models and explain the need for variational approximation of the filtering distribution. The section that follows shows how variational estimates of the filtering distribution can be obtained efficiently. The next section highlights the importance of modeling temporal dependencies and shows how this can be achieved by additionally modeling the difference signal of the aggregate observations. We then present experiments/results on the REDD dataset and conclusions.

Factorial Hidden Markov Models and Variational Inference

FHMMs are a generalization of Hidden Markov Models in which multiple hidden states evolve independently in parallel. See Figure 1c for a representation of the associated graphical model. When the parameters of the individual HMM chains are known, energy disaggregation can be posed as the decoding problem for FHMMs. However, obtaining these paramaters is usually prohibitively expensive. On the other hand, unsupervised energy disaggregation can be posed as the learning problem on this graphical model.

Because of its factorial nature, the probability of the observation at time t, x_t, is a function of the joint hidden state z_t, with $t \in \{1, ..., T\}$. In general, the latent variables of FHMMs are modeled as categorical variables, which could lead to computationally tractable solutions (e.g., (Jang, Gu, and Poole 2016)). In our case we restrict z_t to be binary, specifically Bernoulli distributed. Assuming binary hidden states removes some of the ambiguity inherent to the learning problem: every latent representation with categorical variables can be decomposed into a representation with binary variables, i.e. by assigning a binary variable for each categorical state. This binary decomposition is unique and in a sense maximal. However, binary decompositions can be aggregated into exponentially-many categorical decompositions, i.e. any combination of binary latent variables can be joined into one categorical variable. In order to avoid this ambiguity, we restrict the latent variables to be binary.

Since hidden states are assumed to be binary and multiple hidden states evolve in parallel, $z_t \in Z = \{0, 1\}^C$ with C being the number of parallel hidden chains. Thus, the joint likelihood can be expressed as:

$$p(x_{1:T}, z_{1:T}) = \prod_t^T p(x_t|z_t) \prod_i^C p(z_{t,i}|z_{t-1,i}) p(z_{0,i}) \quad (1)$$

For the application of energy disaggregation, we choose a representation of the aggregate observation similar to our prior work (Lange and Bergés 2016a), i.e. the observation x_t constitutes the aggregate instantaneous power waveform aligned by zero-crossings detected in the voltage line, thus $x_t \in \mathbb{R}^N$ with N being the number of samples per voltage cycle. Figure 1a shows aggregate instantaneous power waveforms alongside the observed aggregate active power over time. Since instantaneous power is additive, we model $p(x_t|z_t)$ to be a Gaussian distribution with $p(x_t|z_t) = \mathcal{N}(x_t|Wz_t, \alpha I)$ where α is a variance parameter, $W \in \mathbb{R}^{N \times C}$ is a matrix containing the power waveforms of the inferred components and is not assumed to be known. Figure 1b shows an example of inferred power waveforms.

Baum-Welch, an Expectaction-Maximization algorithm, is a prominent algorithm for the learning problem in Hidden Markov Models. Baum-Welch makes model updates based on the expected time spent in states, the expected number of state transitions and the expected number of times a state emits an observation. An efficient algorithm to compute these quantities is the forward-backward algorithm. The forward-probabilities (2) can be computed recursively. Given the forward probabilities, the filtering distribution can be computed according to (3).

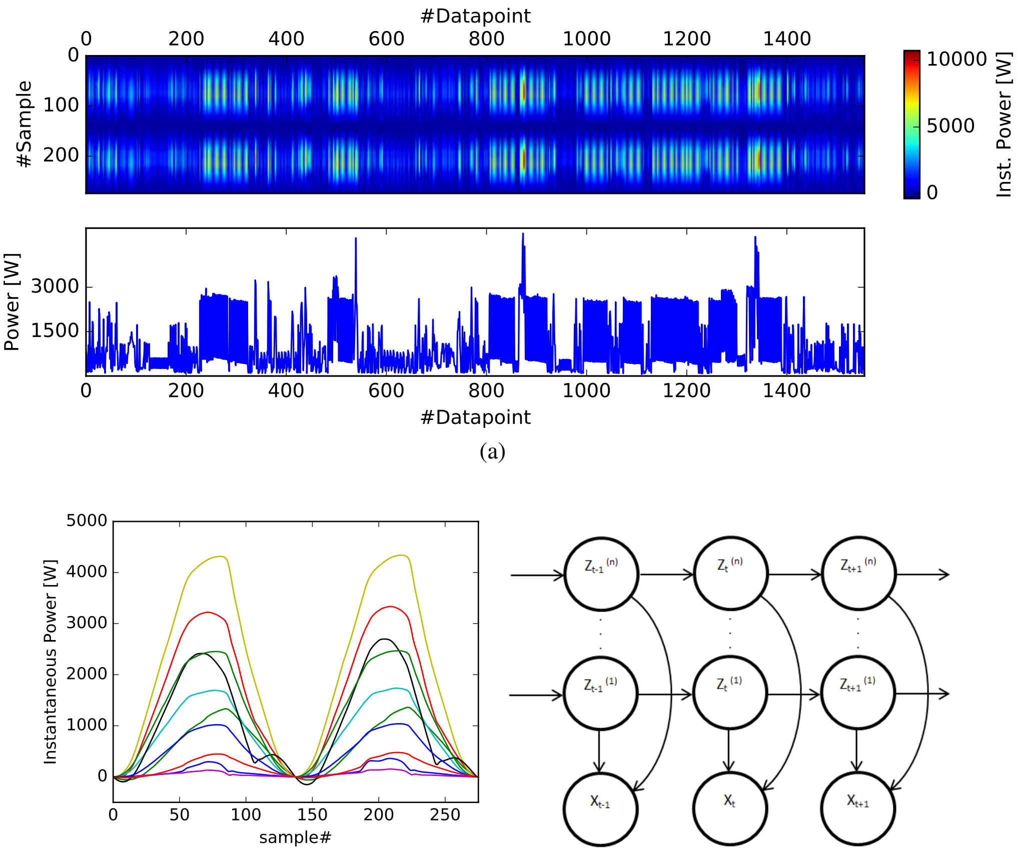
$$p(x_{1:t}, z_t) = p(x_t|z_t) \sum_{z' \in Z} p(z_t|z') p(z_{t-1}, x_{1:t-1}) \quad (2)$$

$$p(z_t|x_{1:t}) = \frac{p(x_{1:t}, z_t)}{\sum_{z' \in Z} p(x_{1:t}, z')} \quad (3)$$

Because the number of possible latent states z grows exponentially with the number of components, evaluating (2) is intractable for FHMMs. However, as we will show later, ideas from Variational Inference can be used to approximate forward-probabilities.

Variational Inference is a tool to deal with intractable posterior distributions and relies on an auxiliary or variational distribution Q governed by the variational parameter Θ. Posterior inference in Q is required to be tractable, which is usually achieved by making independence assumptions. To paraphrase the main idea behind Variational Inference: in order to perform inference on a distribution P with intractable posterior, variational parameters Θ are chosen in such a way that Q best approximates P and then inference is performed on Q instead of P. For our application, P is the filtering distribution, i.e. (3), and we choose Q to be an independent multivariate Bernoulli distribution with density $q_\sigma(z_t) = \prod_i^C \sigma_i^{z_{t,i}} (1-\sigma_i)^{1-z_{t,i}}$ and with σ_i being the coin-flip probabilities of the latent variables. For the conditional $q_\sigma(z_t|x_t)$, we assume the coin-flip probabilities to be functions of the conditioning variable x_t, i.e. $q_\sigma(z_t = \vec{1}|x_t) = \sigma_t = f_\Theta(x_t)$. Because we want Q to capture the temporal dependencies present in P, we chose f to be a recurrent deep neural network governed by the variational parameters Θ (which in this case constitute the weights of the neural network). This in turn means that $\sigma_t = f_\Theta(x_{1:t})$ is a function of all previous observations $x_{1:t}$, i.e. Q is also a filtering distribution: $q(z_t|x_{1:t})$. Note that this implies that the auxiliary distribution does not assume temporal independence between latent variables but assumes independence between elements of the latent variable at any given time.

The evidence lower bound (ELBO) as a variational objective



(a)

(b)

(c)

Figure 1: (a): Matrix X containing aggregate instantaneous power waveforms alongside the aggregate power. (b): Matrix W containing candidate component waveforms. (c): Representation of a Factorial Hidden Markov model

can be derived as follows (Blei 2011):

$$\mathcal{L} = \log p(x_{1:t}) - D_{KL}(q(z_t|x_{1:t})||p(z_t|x_{1:t})) \qquad (4)$$
$$= \mathbb{E}_Q[\log p(x_{1:t}, z_t)] - \mathbb{E}_Q[\log q(z_t)] \qquad (5)$$

Note that because of the equivalence of (4) and (5), maximizing (5) is equivalent to maximizing (4). This means that optimizing the parameters of P and Q w.r.t. to (5) leads to maximization of the log-likelihood of the data as well as minimization of the posterior divergence. Although the second expectation of (5) usually has an analytical solution, evaluating the first expectation is usually achieved by sampling from Q. However, approximating the expression by sampling disconnects the optimization problem from the variational parameters Θ, i.e. Q vanishes from the optimization problem. For some continuous non-conjugate distributions, this problem can be avoided by the re-parameterization trick, i.e. by finding a deterministic and differentiable function that provides samples of Q given the variational parameters Θ and some random noise. For binary non-conjugate distributions such as the Bernoulli such a function does not seem to exist (Kingma and Welling 2013). We circumvent this problem by approximating the true filtering distribution, i.e. estimate $\hat{p}(z_t|x_{1:t})$ with the help of Q. Given estimates $\hat{p}(z_t|x_{1:t})$, the σ^* as a function of Θ resulting in the lowest forward KL-divergence can be obtained, i.e. $\sigma^* = \arg\min_\sigma \sum_t D_{KL}(\hat{p}(z_t|x_{1:t})||q_\sigma(z_t|x_{1:t}))$. Given σ^*, the binary cross-entropy loss between σ and σ^* ($H(\sigma_t, \sigma_t^*)$) is then minimized in order to minimize the forward KL-divergence of the posterior.

Minimizing the binary cross-entropy loss between σ and σ^* minimizes the posterior divergence but since the parameters of P (the component waveforms W) are not known, the model needs to be forced to explain the aggregate signal explicitly. Otherwise the free parameter W will be abused to minimize the divergence without explaining the data. Thus, we additionally maximize $\mathbb{E}_Q[\hat{p}(z_t, x_{1:t})]$ in order to explain the aggregate waveforms. Hence, the objective function becomes:

$$L(\sigma, W) = \sum_t \mathbb{E}_Q[\hat{p}(z_t, x_{1:t})] - H(\sigma_t, \sigma_t^*)$$
$$\text{with: } \sigma_t^* = \min_\sigma D_{KL}(\hat{p}(z_t|x_{1:t})||q(z_t|x_{1:t}))$$
$$= \mathbb{E}_{\hat{p}(z_t|x_{1:t})}[z]$$
$$\text{and: } -H(\sigma_t, \sigma_t^*) = \sigma_t^* \log(\sigma_t) + (1 - \sigma_t^*) \log(1 - \sigma_t)$$

To sum up the main ideas of this paper:

- Computing the filtering recursion required for the E-step of Baum-Welch is prohibitively expensive since it requires a summation over exponentially-many latent configurations.
- An auxiliary distribution $q(z_t|x_{1:t})$ that assumes independence between elements in z_t is introduced, which allows for approximating $\hat{p}(z_t|x_{1:t})$. Note that q does not assume independence over time steps. The independence structure of the auxiliary distribution is then exploited to approximate the filtering recursion efficiently circumventing summation of exponentially-many latent configurations.
- In order to optimize the parameters of the auxiliary distribution, the variational parameters minimizing the KL-divergence between P and Q, i.e. σ^*, are estimated and the binary cross-entropy loss between the predicted parameters σ and the optimal σ^* is minimized. As we will show later, this is equivalent to minimizing the KL-divergence but circumvents the re-parameterization trick and allows for a compact representation of the problem.

Note that we interchangeably call σ and Θ variational parameters. However, in reality, σ is a function of the true variational parameters, i.e. the neural network weights Θ, and when a loss is defined w.r.t. σ gradients w.r.t. Θ can be obtained by application of the chain rule, i.e. back-propagation. In the next section, we will discuss how to obtain estimates of the filtering distribution probabilities, i.e. $\hat{p}(z_t|x_{1:t})$.

Estimating filter distribution probabilities

When using the forward-algorithm to obtain the filtering distribution $p(z_t|x_{1:t})$ for FHMMs, the computational complexity is in $\mathcal{O}(2^{2C}T')$ with C being the number of components and T' being the number of discrete time steps. In this work, we propose a learning algorithm that operates in $\mathcal{O}(C^{\epsilon+1}T)$ with C^{ϵ} being the number of candidate latent state configurations being considered and $T \ll T'$ (by reducing decision variables) and $\epsilon < C$ (by enforcing sparsity).

Reducing decision variables

For the problem of energy disaggregation, the aggregate observation is highly non-iid, i.e. instantaneous power waveforms tend to repeat themselves over time since they are associated with the operational state of appliances (and these do not change very often). This implies that, as long as the aggregate observations have not changed significantly, the latent states will not have changed and no new decision needs to be made. Thus, by employing a simple change-point detector that extracts points in time, so called events, where a significant change in the aggregate power was observed, the number of decision variables can be reduced significantly. Let the number of detected events be T. Reducing the number of decision variables reduces the complexity to $\mathcal{O}(2^{2C}T)$. Depending on the change point detector, T is often three orders of magnitudes smaller than T'.

Enforcing sparsity

A portion of the latent space can be excluded by enforcing sparsity of the latent variables. Usually only a small number of appliances are active at any given time. Thus latent configurations where more than ϵ components are active can be excluded. Let $\mathcal{Z} = \{z \in \{0,1\}^C \,|\, \sum_i z_i < \epsilon\}$ be the set of sparse candidate latent configurations. We assume that $p(x_{1:t}, z_t = z_i) = 0$ for all $z_i \notin \mathcal{Z}$. This assumption allows us to evaluate $p(z_t|x_{1:t})$, since the denominator of equation (3) has become tractable. This assumption reduces the complexity to $\mathcal{O}(C^{2\epsilon}T)$ with $|\mathcal{Z}| \in \mathcal{O}(C^{\epsilon})$.

Variational approximation of $p(z_t|x_{1:t})$

For many problems modeled with FHMMs, such as energy disaggregation, $p(z_t|x_t)$ and therefore $p(z_t|x_{1:t})$ are highly multi-modal distributions. However, since the auxiliary conditional distribution Q assumes independence between elements of z, Q is unable to learn the multi-modality of $p(z|x)$. However, Q is able to either learn $\arg\max_z p(z|x)$ or $\mathbb{E}_{p(z|x)}[z]$. These two modeling choices are reflected in either minimizing the forward $D_{KL}(P||Q)$ or reverse $D_{KL}(Q||P)$, respectively. It can be shown that:

$$\arg\min_{\sigma} D_{KL}(P||Q_\sigma) = \mathbb{E}_{p(z|x)}[z]$$
$$\arg\min_{\sigma} D_{KL}(Q_\sigma||P) = \arg\max_{z} p(z|x)$$

Viterbi learning was proposed as a faster alternative to Baum-Welch. For Viterbi learning the model parameters are updated based on the most probable path $z^*_{1:T} = \arg\max_{z_{1:T}} p(z_{1:T}|x_{1:T})$.

Since minimizing the reverse KL-divergence forces Q to learn the most probable mode of P, minimizing the reverse KL-divergence approximates Viterbi learning but tends to underfit considerably. On the other hand, minimizing the forward KL-divergence seems to preserve more information about state posterior probabilities. However, as we will show later, our proposed method does not correspond fully to learning like Baum-Welch, i.e. updating the model based on $p(z_t|x_{1:\boldsymbol{T}})$, but rather updating based on $p(z_t|x_{1:\boldsymbol{t}})$, that is, making model updates based on forward-probabilities alone whilst ignoring backward-probabilities.

As discussed earlier, the Baum-Welch algorithm as well as Viterbi learning require computations that are quadratic in the numbers of hidden states. Even with the domain-specific sparsity assumptions introduced earlier, computations that are quadratic in the number of latent configurations are still prohibitively expensive. The key insight into circumventing these computations is the fact that the filtering distribution at time t, i.e. $p(z_t|x_{1:t})$ can be approximated by exploiting the independence structure of Q, i.e. the fact that the auxiliary distribution assumes independence between components at any single point in time. Starting with equation (2):

$$
\begin{aligned}
&p(x_{1:t}, z_t)\\
&= p(x_t|z_t) \sum_{z' \in \mathcal{Z}} p(z_t|z') p(z_{t-1} = z', x_{1:t-1})\\
&\approx p(x_t|z_t) \sum_{z' \in \mathcal{Z}} p(z_t|z') \boldsymbol{q}(z_{t-1} = z'|x_{1:t-1}) p(x_{1:t-1})\\
&= p(x_t|z_t) \sum_{z' \in \mathcal{Z}} p(z_t|z') \prod_i \sigma_{t-1,i}^{z'_i} (1 - \sigma_{t-1,i})^{1-z'_i} p(x_{1:t-1})
\end{aligned}
$$

Note that FHMMs components switch independently, i.e. $p(z|z') = \prod_i p(z_i|z'_i)$ and let $\pi(m, n)$ be the state-transition probabilities. Because $q(z_t|x_{1:t})$ assumes independence between elements of z_t, we can simplify the expression by recursively pulling out elements of $q(z_t|x_{1:t})$, ultimately allowing us to rewrite a sum over all possible z into a sum over the number of components, i.e. circumventing computations that grow exponential with the number of parallel latent states:

$$
\begin{aligned}
&p(x_{1:t}, z_t)\\
&\approx p(x_t|z_t) p(x_{1:t-1})\\
&\sum_{z' \in Z} \prod_i p(z_{t,i} = z_i | z_{t-1,i} = z'_i) \sigma_{t-1,i}^{z'_i} (1 - \sigma_{t-1,i})^{1-z'_i}\\
&= p(x_t|z_t) p(x_{1:t-1})\\
&\sum_{z' \in Z} \prod_i (z_i \pi(1, z'_i) + (1 - z_i) \pi(0, z'_i)) \sigma_{t-1,i}^{z'_i} (1 - \sigma_{t-1,i})^{1-z'_i}\\
&= p(x_t|z_t) p(x_{1:t-1})\\
&\sum_i z_i \sigma_{t-1,i} \pi(1,1) + z_i (1 - \sigma_{t-1,i}) \pi(1,0)\\
&+ (1 - z_i) \sigma_{t-1,i} \pi(0,1) + (1 - z_i)(1 - \sigma_{t-1,i}) \pi(0,0)\\
&= \hat{p}(x_{1:t}, z_t)
\end{aligned}
$$

This allows us to approximate the forward probabilities $p(z_t, x_{1:t})$ based on σ_{t-1} as provided by Q and W as a parameter of P. Since we can compute $\hat{p}(z_t, x_{1:t})$ for all sparse $z \in \mathcal{Z}$, we can approximate the filtering distribution $\hat{p}(z_t|x_{1:t})$ according to equation (3). Note that, since we are only interested in the filtering distribution, $p(x_{1:t-1})$ does not need to be modeled because it cancels out.

Let $\sigma^* = \mathbb{E}_{\hat{p}(z_t = z|x_{1:t})}[z]$. Even though σ^* is a function of W, we treat σ^* as a constant and do not allow the gradient of W to flow into σ^*. This avoids W being exploited to minimize the posterior divergence instead of explaining the aggregate data. Note that, when allowing the gradient of W to flow into σ^*, the algorithm will infer nonsensical component waveforms, i.e. waveforms that draw significant power when the voltage is 0. Based on the same reasoning, we do not allow the gradient of σ to flow into $\mathbb{E}_Q[\hat{p}(z_t, x_{1:t})]$.

Thus by exploiting the independence assumption of the auxiliary distribution, the computational complexity estimating the filtering distribution can be reduced to $\mathcal{O}(C^{\epsilon+1}T)$.

There is at least one example in the literature showing an application of variational inference for learning in FHMMs: in (Ng, Chilinski, and Silva 2016) Gaussian copulas are paired with variational inference to minimize an objective including the reverse KL-divergence, thus circumventing the problem of having to approximate the filtering distribution. Furthermore, although not applied to sequential data and therefore not modeling temporal dependencies between latent variables, previous work has proposed approaches for the subproblem of estimating the gradient through binary stochastic units, e.g. (Raiko et al. 2014; Bengio, Léonard, and Courville 2013). When temporal dependencies are removed, (Tang and Salakhutdinov 2013) arrive at a similar solution to ours. Their respective loss is derived as: $L_{T\&S} = \sum_m \overline{w}^{(m)} [\log p(x|z^{(m)}) + \log p_\sigma(z^{(m)}|x)]$ with $\overline{w}^{(m)}$ being normalized importance weights of configuration $z^{(m)}$. Note that, if the sparsity constraints introduced here were to be applied, the importance weights would approximate $p(z|x)$. Also note that in that case, even though motivated differently, the gradient updates w.r.t. the component activation probabilities σ are equivalent when temporal dependencies are not modeled. This justifies the seemingly arbitrary choice of minimizing the cross-entropy loss, i.e. $H(\sigma, \sigma^*)$ (instead of e.g. $(\sigma - \sigma^*)^2$).

It can be shown that:

$$
\begin{aligned}
\frac{\partial L_{T\&S}}{\partial \sigma_i} &= \frac{1}{-(1 - \sigma_i)} [1 - \sigma_i^*] + \frac{1}{\sigma_1} [\sigma_i^*]\\
&= \frac{\partial (1 - \sigma_i^*) \log(1 - \sigma_i) + \sigma_i^* \log(\sigma_i)}{\partial \sigma_i}\\
&= \frac{\partial H(\sigma, \sigma^*)}{\partial \sigma_i}
\end{aligned}
$$

Modeling temporal dependencies

Building on experience from previous work (Lange and Bergés 2016a), the main objective of modeling the temporal dependencies between latent states is temporal regularization. Specifically for the problem of energy disaggregation, this means that when a single appliance changes its state, only one and not multiple components change state. Without modeling the temporal dependencies, models tend to 'stitch', i.e. when a single appliance turns on, multiple model components switch states. Also without modeling temporal dependencies, the model 'recycles' components, e.g. appliance a might be explained by components 1 and 2, then appliance b is explained by components 2 and 3 and appliance c is explained by components 1 and 3. A linear mapping from components to appliances then becomes impossible.

Furthermore, for energy disaggregation, introducing fixed state transition probabilities is problematic because of vast differences in the power consumption of appliances. When every component pays a fixed cost for switching ($\pi(0, 1)$ or $\pi(1, 0)$), appliances with a high power consumption can still afford to be explained by multiple components because the cost for under-estimating the aggregate is higher than multiple switching costs. At the same time, appliances that consume little power will be ignored, since when they turn on, the associated increase in aggregate loss does not outweigh the switching cost.

To overcome this problem, we additionally model the difference signal $\delta x_t = x_t - x_{t-1}$ similar to (Kolter and Jaakkola 2012). Note that although technically the graphical model

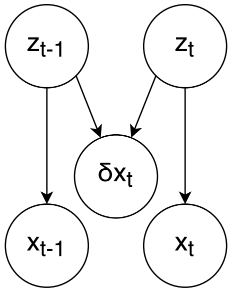


Figure 2: Graphical Model when additionally modeling the difference signal δx_t

changes (see Figure 2), π can also be viewed as a function of δx_t, i.e. the switching probabilities depend on how well each component explains δx_t. We define switching probabilities associated with each component turning *on* or *off* at time t. Additionally, we define a switching probability associated with no component switching.

Let,

$$\begin{aligned} \mathcal{I}(t,i) &= exp[-\beta||W_{i:} - \delta x_t||] && \text{(on-switch)} \\ \mathcal{O}(t,i) &= exp[-\beta||W_{i:} + \delta x_t||] && \text{(off-switch)} \\ \mathcal{X}(t) &= exp[-\beta||\delta x_t||] && \text{(no-switch)} \end{aligned}$$

Following the intuition gained earlier, we estimate the filtering distribution as,

$$\begin{aligned} &\hat{p}(z_t = z, x_{1:t}) = p(x_t|z_t = z)p(x_{1:t-1}) \\ &[\sum_i^C(\sigma_{t-1,i}(1-z_i)\mathcal{O}(t,i) + (1-\sigma_{t-1,i})z_i\mathcal{I}(t,i)) \\ &\quad + \prod_i^C(z_i\sigma_{t-1,i} + (1-z_i)(1-\sigma_{t-1,i}))\mathcal{X}(t)] \end{aligned} \tag{6}$$

Note that the model described in equation (6) models dependencies between components to some degree. The product in the last line can be expanded into all combinations of component configurations where no component switches from $t-1$ to t. This factorization allows for a compact and differentiable representation of 'no component'-switches without having to enumerate an exponential number of configurations, therefore modeling limited dependencies between components efficiently.

Resulting Algorithm: Variational BOLT

The resulting algorithm, which we call Variational BOLT, operates in temporal mini-batches of a fixed time-horizon h, i.e. the data is sequentially fed into the neural network and model parameters Θ and W are updated before a new mini-batch of data is processed. This process is repeated until convergence. Algorithm 1 explains the process in pseudo-code.

The resulting algorithm has similarities to Variational Autoencoders (VAE) as well as Expectation Maximization, specifically the Baum-Welch algorithm. Like VAE, an efficient auxiliary recognition distribution is trained to predict the parameters of the latent distribution. However, the auxiliary distribution is solely used to speed up computations of the filtering recursion. Unlike VAE and like EM, instead of approximating intractable expectations by sampling latent states from the recognition distribution, updates are computed based on a fixed set of possible hidden states.

Algorithm 1: Variational BOLT in pseudo-code

```
input  : Dataset $X$ of size $T \times N$
output: Trained model parameters $W$ and $\Theta$
Initialize $W$ by clustering and $\Theta$ randomly;
while not converged do
    $t \leftarrow 0$;
    $\sigma_0 \leftarrow \vec{0}$;
    while $t < T$ do
        for $t' \in t : t + h$ do
            Neural Network forward pass;
            $\sigma_{t'} = f_\Theta(X[1 : t', :])$;
            for $z \in \mathcal{Z}$ do
                Compute $\hat{p}(x_{1:t'}, z)$ based on (6);
            for $z \in \mathcal{Z}$ do
                Compute $\hat{p}(z|x_{1:t'})$ based on (3);
            $\sigma^*_{t'} \leftarrow \mathbb{E}_{\hat{p}(z|x_{1:t'})}[z]$;
        Maximize $\sum_{t'=t}^{t+h} H(\sigma_{t'}, \sigma^*_{t'})$ w.r.t. $\Theta$;
        Maximize $\sum_{t'=t}^{t+h} \mathbb{E}_Q[\hat{p}(z, x_{1:t'})]$ w.r.t. $W$;
        $t \leftarrow t + h$;
```

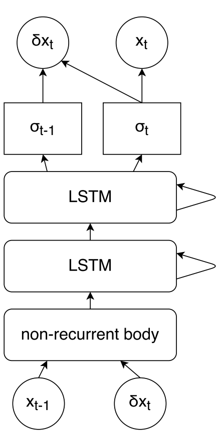


Figure 3: Topology of the neural network

Experiments

Experiments were conducted on the publicly-available REDD (Kolter and Johnson 2011) dataset. The dataset contains current and voltage readings at the main distribution panel with a sampling rate of 16kHz and breaker level power readings with a sampling frequency of 0.3Hz. The neural network used to predict $q(z_t = \vec{1}|x_{1:t})$ is a 4 layer recurrent neural network. The bottom two layers constitute non-recurrent $tanh$ layers with 200 output units each. The top two layers are LSTM-layers with $sigmoid$-activations each

(a) Circuit	AFAMAP (supervised*)	VarBOLT (unsupervised)
Microwave	97.5% / 66.1%	88.8% / 8.0%
Bath GFI	82.7% / 70.8%	71.9% / 40.2%
Electronics	41.6% / 0.8%	87.8% / 40.7%
Kitch. Out. 1	37.5% / 12.9%	8.6% / 32.8%
Furnace	91.7% / 70.8%	85.0% / 50.6%
Kitch. Out. 2 .	45.2% / 16.0%	5.3% / 70.1%
Washer/Dryer	98.8% / 73.6%	97.3% / 72.3%

(b)	NFHMM (unsupervised)	VarBOLT (unsupervised)
Overall panel	0.25	0.63

Table 1: (a) Performance comparison to AFAMAP, a supervised inference technique paired with an unsupervised strategy of obtaining ground truth. Performance is measured in Precision / Recall. (b) Performance comparison with NFHMM, another end-to-end unsupervised approach, in GSPA.

with 100 and 10 output units respectively. This means that 10 components were extracted and maximally 6 out of these 10 inferred components were allowed to be active at any given time ($\epsilon = 6$). Figure 3 shows a graphical depiction of the neural network.

Change points of the aggregate power were detected by an event detection algorithm: Let $p(t)$ be the aggregate power at time t. The maximum value of the absolute difference in the power signal within a window of 5 time steps was extracted. Every window then casts a vote for the highest absolute power difference. However, only these timestamps for which $|p(t) - p(t-1)| > 50W$ holds can receive a vote. Every time stamp that received more than 3 votes is considered an event. Then, in order to reduce the number of decision variables, the mean instantaneous power waveforms in between events was extracted, and these constitute the set of T values of x_t.

The neural network was then fed x_t and $\delta x_t = x_t - x_{t-1}$ and tasked to explain x_{t+1} and δx_t. In order to speed up convergence, the appliance waveforms W were initialized by the cluster centroids obtained by applying K-Means to the difference signal δx_t. In the experiments the hyper-parameters α and β, i.e. the variance of the difference and aggregate model were kept at 1. The model was trained for 200 iterations. For inference, the filtering distribution probabilities were simply binarized: $z = \sigma > 0.5$.

Results

Since appliances were sub-metered at the circuit level and some circuits contain multiple appliances, precision and recall are used as a metric. “Recall measures what portion of a given circuits energy is correctly classified, while precision measures, of the energy assigned to a circuit, how much truly belonged to that circuit” (Kolter and Jaakkola 2012). For every pair of inferred component and circuit, precision and recall were computed and the component resulting in the highest $(prec + recall)/2$ was selected for this circuit. Note that because we assume z to be binary, we implicitly assume appliances to be 2-state, i.e. they can either be *on* or *off*. However, appliances like e.g. a furnace are composed of multiple sub-elements. In that case, the proposed model ‘over-disaggregates’, i.e. it assigns a component for every sub-element. An example of ‘over-disaggregation’ can be seen in Figure 4. Furthermore, some appliances have different power levels according to their operational state, i.e. a hair-dryer has different heat settings. In this case, the proposed methods assigns different components for the same appliances. Note that supervised inference techniques usually do not suffer from these problems. This is why we expect supervised inference algorithms to outperform our approach. Table 1(a) shows a comparison with AFAMAP (Kolter and Jaakkola 2012). AFAMAP is a supervised inference algorithm paired with an unsupervised strategy of obtaining model parameters for the individual HMM chains. We also compare the performance to a fully unsupervised method based on Non-parametric FHMMs (NFHMM) proposed in (Jia, Gao, and Spanos 2015) (1(b)). As a performance criterion, they propose GSPA (worst 0 - 1 best). GSPA does not measure differences in power but rather differences between activations, i.e. circuit power traces are binarized and then GSPA measures a weighted ratio between the intersection and union between binarized ground thruth

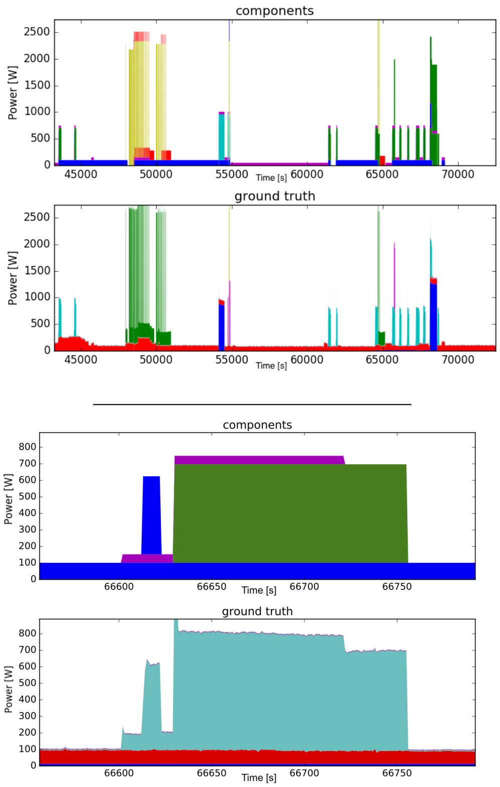


Figure 4: Top: Inferred components as well as circuit level ground truth. Bottom: An example of ‘over-disaggregation’ of the furnace.

and estimates.

Conclusion & Future Work

We proposed a variational learning algorithm for discrete Factorial Hidden Markov Models and applied it to the problem of energy disaggregation. The algorithm compares promisingly to a supervised inference algorithm that is paired with an unsupervised approach of obtaining ground truth. When compared to another end-to-end unsupervised approach, our proposed method significantly outperforms it. An implementation in *keras* (Chollet 2015) can be found at: *https://github.com/INFERLab/varbolt*. Once the auxiliary distribution is trained, the corresponding neural network could in principle be deployed to sensing hardware located at the electrical panel for on-premise real-time inference.

Furthermore, we believe that our proposed method opens many interesting research paths as there is still much room for improvement. A possible research path is to combine our methods with ideas from (Tang and Salakhutdinov 2013). By sampling candidate hidden configurations, the sparsity constrains made in Section 3 can, in principle, be relaxed. This may allow the model to scale up to more components while keeping computational cost low. Furthermore, the current model for the difference signal uses the Euclidean distance to judge the similarity between component waveforms and the difference signal, so investigating other similarity measures could further refine the model since Euclidean distances might overemphasize differences in power over differences in the shape of the waveform.

References

Asif, H. S., and Sanguinetti, G. 2011. Large-scale learning of combinatorial transcriptional dynamics from gene expression. *Bioinformatics* 27(9):1277–1283.

Bengio, Y.; Léonard, N.; and Courville, A. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Blei, D. M. 2011. Variational inference. *Lecture from Princeton, https://www.cs.princeton.edu/courses/archive/fall11/cos597C/lectures/variational-inference-i.pdf*.

Chollet, F. 2015. keras. https://github.com/fchollet/keras.

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.

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

Jang, E.; Gu, S.; and Poole, B. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Jia, R.; Gao, Y.; and Spanos, C. J. 2015. A fully unsupervised non-intrusive load monitoring framework. In *Smart Grid Communications (SmartGridComm), 2015 IEEE International Conference on*, 872–878. IEEE.

Kingma, D. P., and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

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.

Lange, H., and Bergés, M. 2016a. Bolt: Energy disaggregation by online binary matrix factorization of current waveforms. In *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, 11–20. ACM.

Lange, H., and Bergés, M. 2016b. Efficient inference in dual-emission fhmm for energy disaggregation. In *Workshops at the Thirtieth AAAI Conference on Artificial Intelligence*.

Ng, Y. C.; Chilinski, P. M.; and Silva, R. 2016. Scaling factorial hidden markov models: Stochastic variational inference without messages. In *Advances in Neural Information Processing Systems*, 4044–4052.

Raiko, T.; Berglund, M.; Alain, G.; and Dinh, L. 2014. Techniques for learning binary stochastic feedforward neural networks. *arXiv preprint arXiv:1406.2989*.

Stanculescu, I.; Williams, C. K.; and Freer, Y. 2014. A hierarchical switching linear dynamical system applied to the detection of sepsis in neonatal condition monitoring. In *UAI*, 752–761.

Tang, Y., and Salakhutdinov, R. R. 2013. Learning stochastic feedforward neural networks. In *Advances in Neural Information Processing Systems*, 530–538.