A NOVEL DNN-HMM-BASED APPROACH FOR EXTRACTING SINGLE LOADS FROM AGGREGATE POWER SIGNALS

Lukas Mauch and Bin Yang

Institute of Signal Processing and System Theory, University of Stuttgart, Germany

ABSTRACT

This paper presents a new supervised approach to extract the power trace of individual loads from single channel aggregate power signals in non-intrusive load monitoring (NILM) systems. Recent approaches to this source separation problem are based on factorial hidden markov models (FHMM). Drawbacks are the needed knowledge of HMM models for all loads, what is infeasible for large buildings, and the large combinatorial complexity. Our approach trains HMM with two emission probabilities, one for the single load to be extracted and the other for the aggregate power signal. A Gaussian distribution is used to model observations of the single load whereas observations of the aggregate signal are modeled with a Deep Neural Network (DNN). By doing so, a single load can be extracted from the aggregate power signal without knowledge of the remaining loads. The performance of the algorithm is evaluated on the Reference Energy Disaggregation (REDD) dataset.

Index Terms— Non-intrusive load monitoring (NILM), supervised power disaggregation, Hidden Markov Model (HMM), Deep Neural Networks (DNN)

1. INTRODUCTION

Non-intrusive load monitoring is an energy monitoring technology which has attracted much attention in the recent years. The aim is to estimate the power consumed by individual loads if only the aggregate power signal is measured with one power meter [1]. In terms of signal processing, NILM is equivalent to a single channel source separation problem where the power consumed by one load corresponds to one mixture component of the sum signal [2].

The load and aggregate signals are power signals and therefore nonnegative. Depending on the size of the building, the aggregate signal may contain only a few or up to hundreds of load components. The components are non-stationary and exhibit strong temporal relations ranging from seconds to many hours, depending on how long a load is active. Considering only on/off and multistate devices, load signals are piecewise constant and distinguishable only by amplitude and location of change points. Because of these properties, well established techniques from audio source separation like Independent Component Analysis (ICA) or Non-negative Matrix Factorization (NMF) fail if applied to the NILM problem. ICA needs multiple measurements which are not available. NMF is suited for part based decomposition making no assumptions about their statistical dependencies [3]. It works best if signal parts naturally cluster and therefore is not suited for highly correlated load signals.

Most common approaches to solve the NILM problem are based on unsupervised event detection [4, 5, 6, 7], splitting the aggregate signal into piecewise constant parts. Then the aggregate signal, features are extracted from the detected events and are used to assign them to individual loads [8, 9, 10]. The problem to find reliable event detection and classification methods is still an ongoing research and is not yet solved satisfactory. Another severe drawback of the event based approaches is that they only make little use of the strong temporal relations between the events.

Recently, supervised eventless methods making use of these temporal relations came into focus. The most popular method is based on FHMM [11]. The aggregate signal is assumed to be a sum of observation sequences generated by multiple HMMs [12, 13, 14]. The combination of state sequences that explains the aggregate signal best is found by solving an optimization problem, maximizing the posterior probability of the state sequence combination. This approach is interesting because the obtained state sequences are related to the use cases of the individual loads and could be used for their semantic analysis. The model is generative because future power consumption can be predicted from past observations. However, severe limitations make this approach impractical. First, each load needs a known HMM what is impractical for large buildings. Secondly, even if all HMMs are known, an exact inference of the state sequences is intractable because the number of possible sequence combinations grows exponentially with the sequence length and the number of loads.

This paper proposes a new approach for power disaggregation based on HMM and DNN. For each load to be extracted from the aggregate signal, one HMM is used to model its statistical and temporal properties. Unlike FHMM, the distribution of the aggregate signal is not modeled as a superposition of observations from all HMMs. In our approach, each HMM is augmented with a DNN trained to model the prob-

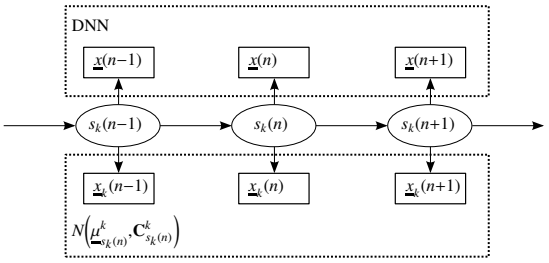

Fig. 1. The DNN-HMM network for signal extraction

ability observing the aggregate signal. The model is trained to maximize the likelihood of simultaneously observing the single load and the aggregate signal. Therefore, the state sequence of the single load can be inferred from the aggregate signal without knowledge of the remaining loads, what is impossible with FHMM.

The paper is organized as follows. In section 2, the hybrid DNN-HMM network is introduced. Section 3 explains the training and inference and Section 4 gives experimental results for the REDD dataset [15].

2. THE DNN-HMM NETWORK

Let $x(n) = \sum_{k=1}^{M} x_k(n)$ be the aggregate signal of M loads $x_k(n) \geq 0$. The signals are divided into non-overlapping blocks $\underline{x}_k(n) = [x_k(nL), \ldots, x_k((n-1)L+1)]^T$ and $\underline{x}(n) = [x(nL), \ldots, x((n-1)L+1)]^T$ of length L, during which the load signal can be assumed to be stationary. In the following, vectors are written with underline and bold font is used to indicate matrices.

2.1. Architecture of the HMM

The used DNN-HMM is shown in Fig. 1. It is an HMM with two emission probabilities per state. One models the probability of observing a single load signal $\underline{x}_k(n)$ and is assumed in this paper to be Gaussian. The second one models the probability of observing the aggregate signal $\underline{x}(n)$. Because this distribution can be very complex, depending on the mixture of loads in the sum signal, a DNN is used for this purpose. Both emission probabilities are linked by the hidden state which corresponds to different use cases of the single load. For a given state sequence $\{s_k(n)\}$, both models are assumed to be statistically independent. The HMM of the k-th load is defined by

$$
\begin{aligned}
s_k(n) &\in \{1, \cdots, M_k\} && (1)\\
\underline{\pi}^k &= [P(s_k(1)=1), \ldots, P(s_k(1)=M_k)]^T && (2)\\
p^k_i &= p(\underline{x}_k(n)|s_k(n)=i) \sim N(\underline{\mu}^k_i, \mathbf{C}^k_i) && (3)\\
\tilde{p}^k_i &= p(\underline{x}(n)|s_k(n)=i) && (4)\\
\mathbf{A}^k &= [P(s_k(n)=i|s_k(n-1)=j)]_{ij}. && (5)
\end{aligned}
$$

Each HMM has M_k discrete states $s_k(n)$ at discrete time n. The initial state probabilities are summarized in $\underline{\pi}^k$. The power consumed at each time instant only depends on the current state and is defined by the Gaussian probability density function (pdf)

$$
p^k_i = \frac{\exp\left(-\frac{1}{2}(\underline{x}_k(n)-\underline{\mu}^k_i)^T \mathbf{C}^{k^{-1}}_i (\underline{x}_k(n)-\underline{\mu}^k_i)\right)}{(2\pi)^{1/2}|\mathbf{C}^k_i|^{1/2}} \qquad (6)
$$

with state dependent but time-constant mean $\underline{\mu}^k_i$ and covariance matrix $\mathbf{C}^k_i$. For extracting the single load state sequence from the aggregate signal, each HMM is augmented with the emission pdf of the aggregate signal $\tilde{p}^k_i = p(\underline{x}(n)|s_k(n)=i)$ that has the parameters $\underline{\theta}^k$, see section 2.2. The current HMM state only depends on the previous one and changes according to the transition probabilities given in the constant transition matrix $\mathbf{A}^k$. All parameters of the augmented HMM are summarized in the vector $\underline{\lambda}_k$, containing all elements of $(\underline{\pi}_k, \mathbf{A}^k, \underline{\mu}^k_i, \mathbf{C}^k_i, \underline{\theta}^k)$ for all $i = 1, \ldots, M_k$.

This HMM defines the joint distribution of the state sequence $\{s_k\} = \{s_k(1), \ldots, s_k(N)\}$, the observation sequence $\{\underline{x}_k\} = \{\underline{x}_k(1), \ldots, \underline{x}_k(N)\}$ and the aggregate sequence $\{\underline{x}\} = \{\underline{x}(1), \ldots, \underline{x}(N)\}$ of length N as

$$
\begin{aligned}
p(\{s_k\}, \{\underline{x}_k\}, \{\underline{x}\}) &= \underline{\pi}^k_{s_k(1)} p^k_{s_k(1)} \tilde{p}^k_{s_k(1)} \cdot \\
&\cdot \prod_{n=2}^{N} p^k_{s_k(n)} \tilde{p}^k_{s_k(n)} \mathbf{A}_{s_k(n), s_k(n-1)}, \qquad (7)
\end{aligned}
$$

where $\underline{\pi}^k_i$ is the i-th element of $\underline{\pi}^k$ and $\mathbf{A}^k_{i,j}$ denotes the (i,j)-th element of $\mathbf{A}$.

2.2. Architecture of the DNN

A DNN is used to estimate the emission pdf $\tilde{p}^k_{s_k(n)}$ of the aggregate signal in each state. At each time n, the DNN gets observations $\underline{x}^{(0)}(n) = [\underline{x}^T(n-B), \ldots, \underline{x}^T(n+B)]^T$ in a context window of length $2B+1$. As shown in Fig. 2, the DNN consists of D layers with $N^{(d)}$ units in the d-th layer. It implements the nonlinear mapping $\underline{x}^{(D)}(n) = [f_1(\underline{x}^{(0)}(n)), \ldots, f_{M_k}(\underline{x}^{(0)}(n))]^T$ with

$$
\begin{aligned}
\underline{a}^{(d)}(n) &= \mathbf{W}^{(d)} \underline{x}^{(d-1)}(n) + \underline{b}^{(d)},\ 1 \leq d \leq D && (8)\\
\underline{x}^{(d)}(n) &= \Phi^{(d)}(\underline{a}^{(d)}(n)) && (9)
\end{aligned}
$$

$\mathbf{W}^{(d)} \in \mathbb{R}^{N^{(d)} x N^{(d-1)}}$ and $\underline{b}^{(d)} \in \mathbb{R}^{N^{(d)}}$ are the weight matrix and bias vector of the d-th layer. The output of elementwise activation function is $\Phi^{(d)}(\cdot)$. For all hidden layers $1 \leq d < D$, rectified linear activation is used. For the ouput layer D, with $N^{(D)} = M_k$ units, a softmax activation is used. Each output unit computes the posterior probability $P(s_k(n)=i|\underline{x}^{(0)}(n)) = f_i(\underline{x}^{(0)}(n))$. Knowing that from the training data, the pseudo likelyhood

$$
\tilde{p}_{s_k(n)} \sim \frac{P(s_k(n)|\underline{x}^{(0)}(n))}{P(s_k(n))} \qquad (10)
$$

can be calculated using the Bayes rule and is used for Eq. 4. The parameter vector $\underline{\theta}^k$ contains all parameters of the DNN of load k, i.e. all elements of $\mathbf{W}^{(d)}$ and $\underline{b}^{(d)}$, for all $d = 1, \ldots, D$.

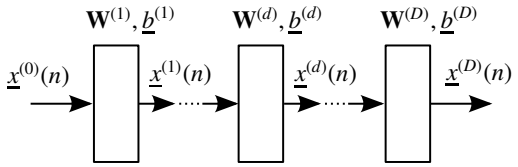

Fig. 2. Architecture of the DNN

2.3. Inference and signal extraction

After the parameters $\underline{\lambda}_k$ have been learned from a training set, the most likely state sequence $\{s_k\} = \{s_k(1), \ldots, s_k(N)\}$ of load k can be infered from observations of the aggregate signal $\{\underline{x}\} = \{\underline{x}(1), \ldots, \underline{x}(N)\}$. During state inference, the load observation sequence $\{\underline{x}_k\}$ is treated as hidden. The state sequence determined by maximizing the conditional distribution

$$P(\{s_k\}|\{\underline{x}\}, \underline{\lambda}_k) = \frac{1}{Z}\underline{\pi}^k_{s_k(1)}\tilde{p}^k_{s_k(1)}\prod_{n=2}^{N}\tilde{p}^k_{s_k(n)}\mathbf{A}_{s_k(n),s_k(n-1)}, \quad (11)$$

obtained by a marginalization of Eq. 7 over all $\{\underline{x}_k\}$ and conditioning on $\{\underline{x}\}$. The partition function Z is independent of $\{s_k\}$ and assures that Eq. 11 sums up to one. The maximum of Eq. 11 is evaluated with the Viterbi algorithm and gives the estimated state sequence $\{\hat{s}_k\}$. For a given $\{\hat{s}_k\}$, the most likely observation sequence $\{\hat{\underline{x}}_k\}$ is estimated by maximizing $p(\{\underline{x}_k\}|\{\hat{s}_k\}) = \prod_{n=1}^{N} p_{s_k(n)}$ which results to the state mean

$$\hat{\underline{x}}_k(n) = \underline{\mu}^k_{s_k(n)}. \quad (12)$$

3. SUPERVISED TRAINING OF THE DNN-HMM NETWORK

The model of each load k is trained separately using a training set $T = (\{\underline{x}^t_k\}, \{\underline{x}^t\})$, containing sequences of observations of the single load k and the aggregate signal. The parameters of the model are chosen to maximize

$$\hat{\underline{\lambda}}_k = \arg\max p(\{\underline{x}^t\}, \{\underline{x}^t_k\}|\underline{\lambda}_k) \quad (13)$$

for any state sequence. This can be done using the prominent Baum-Welch algorithm. To reduce computational complexity, training is split into three parts in our case: a) Maximize Eq. 13 for all possible observations of the aggregate signal $\{\underline{x}^t\}$, i.e. maximize

$$p(\{\underline{x}^t_k\}|\underline{\lambda}_k) = \sum_{\{s_k\}}\underline{\pi}^k_{s_k(1)}p^k_{s_k(1)}\prod_{n=2}^{N}p^k_{s_k(n)}\mathbf{A}_{s_k(n),s_k(n-1)}, \quad (14)$$

using the Baum-Welch algorithm, b) Infer the most likely state sequence $\{\hat{s}^t_k\}$ for given $\{\underline{x}^t_k\}$ for all possible $\{\underline{x}^t\}$ using the Viterbi algorithm, c) Train the parameters of the DNN.

The cost for the DNN training is the negative log-likelihood

$$\begin{aligned} J(\{\underline{x}^t_k\}, \{\underline{x}^t\}) &= -\log\prod_{n=1}^{N} p(\underline{x}^t_k(n), \underline{x}^t(n)|\hat{s}^t_k(n-1), \hat{s}^t_k(n+1)) \\ &= -\sum_{n=1}^{N}\log\left(\sum_{i=1}^{M_k}\mathbf{A}_{i,\hat{s}^t_k(n-1)}\tilde{p}^k_i p^k_i \mathbf{A}_{\hat{s}^t_k(n+1),i}\right). \quad (15) \end{aligned}$$

Knowing the neighbour states $\hat{s}^t_k(n-1)$ and $\hat{s}^t_k(n+1)$ for each instant n, we maximize the likelihood of jointly observing $\underline{x}^t(n)$ and $\underline{x}^t_k(n)$. We use two regularizations

$$J_r(\{\underline{x}^t_k\}, \{\underline{x}^t\}) = J(\{\underline{x}^t_k\}, \{\underline{x}^t\}) + R_{l_2} + R_o \quad (16)$$

The l_2 regularization term $R_{l_2} = \alpha\sum_{d=1}^{D}\|\mathbf{W}^{(d)}\|^2$ is against overfitting. Because load signals are often corrupted with a constant offset due to permanently on loads (e.g. security systems), we use tangent propagation to enforce offset invariance. If the input to the DNN corrupted by a constant offset ξ is $\tilde{\underline{x}}^{(0)} = \underline{x}^{(0)} + \xi\underline{1}$, forcing the output $\underline{x}^{(D)}$ of the network to be constant for all ξ gives the condition

$$\mathbf{J}(\underline{a}^{(1)})\mathbf{W}^{(1)}\underline{1} = \underline{0}, \quad (17)$$

where $\mathbf{J}(\underline{a}^{(1)})$ is the Jacobian matrix of the network output $\underline{x}^{(D)}$ with respect to the activation of the first hidden layer $\underline{a}^{(1)}$. Therefore, $R_o = \beta\|\mathbf{W}^{(1)}\underline{1}\|_1$. It encourages the features learned by the first layer to have zero mean. The parameters are updated using stochastic gradient descent with momentum $\nu = 0.5$ [16], dropout $p_d = 0.5$ [17] and exponential learning rate decay.

4. EXPERIMENTS AND RESULTS

The DNN-HMM algorithm is implemented in Python using Theano and Scikit-learn [18], [19]. All experiments are done with the Reference Energy Disaggregation Dataset (REDD) [15]. It contains real power measurements for six houses. In each house, two aggregate signals of phase A and B with a sampling frequency $1Hz$ and submetered power signals of individual loads with a sampling frequency $1/3Hz$ are recorded. In our study, house 1 with 18 loads and 620 hours of data is used.

Appl.	E_t	$\hat{E}_t$	NASD	Gain
FR	4.30	4.26	0.14	10.5
DW	0.93	0.94	0.08	21.1
MW	1.66	1.49	0.27	13.1
KO	0.35	0.34	0.22	21.0

Table 1. Power based metrics

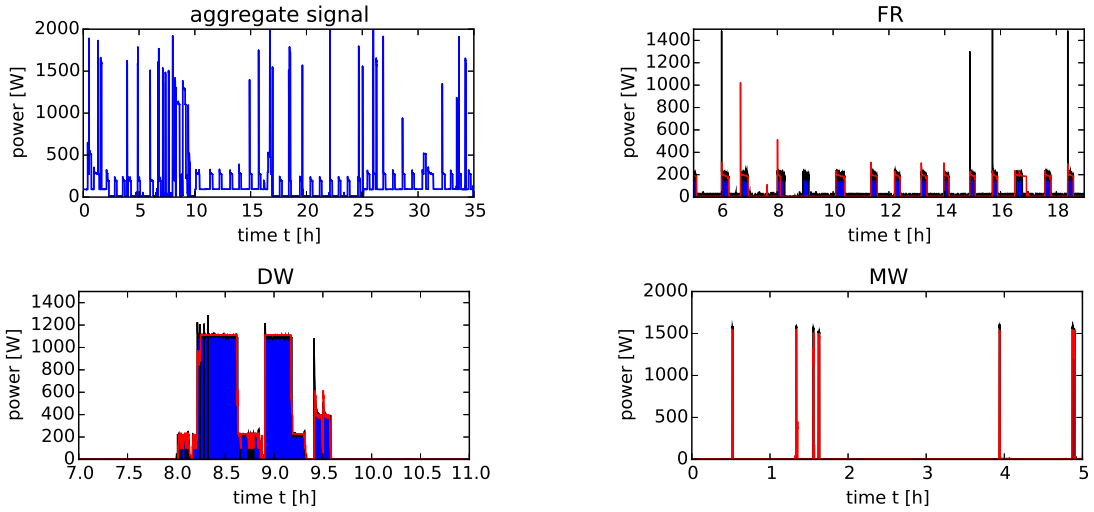

Fig. 3. Aggregate, ground truth (filled blue) and extracted (red) signals.

Because of different sampling frequencies in aggregate and submetered signals, we performed our tests on synthetic aggregate signals by summing up all $1/3Hz$ submetered signals $x_k(n)$. For training and test, both aggregate and submetered signals of house 1 are divided into a training set containing the first 4/5 part (20.7 days) and a test set containing the last 1/5 part (5.2 days).

Four different models are trained to extract the fridge (FR), dishwasher (DW), microwave (MW) and kitchen outlet (KO) from the aggregate signal. The FR is an on/off device with strong periodic behaviour. The DW and MW are multistate devices following similar state sequences. The KO contains on/off devices with arbitrarily occuring on/off events.

In our experiments, we use DNN-HMM models with $M_k = 25$ states. The block length used to extract the observation vectors is chosen to $L = 20$ samples corresponding to one minute of data. The DNN has $D - 1 = 3$ hidden layers with $N^{(d)} = 800$ units each. Its input is a frame of $2B+1 = 41$ blocks corresponding to a time length of 41 minutes. The results are shown in Fig. 3. The first plot gives the aggregate signal, the following plots show the extracted signal (red) of three appliances with the submeter signals as ground truth (filled blue).

In Fig. 3, FR has a low signal amplitude but is frequently active. Except for $8h < t < 10h$, where the signal is mixed with loads of large amplitude, the FR signal is extracted accurately. Position, shape and duration of the active periods fit well to the ground truth. The DW consumes a large power, but is inactive most of the time. Its only active period is at $8h \leq t \leq 9.75h$. It is extracted with similar accuracy as the FR. Even parts where it behaves like a variable load ($9.25h < t < 9.75h$) are correctly approximated. This implies that this approach is not limited to on/off and multistate devices. For MW, position and duration of the active periods is estimated correctly, although there are minor errors in the signal amplitude.

We define the performance metrics.

$$E_k = \frac{1}{F_s}\sum_{n=1}^{N} x_k(n) \tag{18}$$

$$\hat{E}_k = \frac{1}{F_s}\sum_{n=1}^{N} \hat{x}_k(n) \tag{19}$$

$$\mathrm{NAD} = \sqrt{\frac{\sum_{n=1}^{N} |\hat{x}_k(n) - x_k(n)|}{\sum_{n=1}^{N} |x_k(n)|}}, \tag{20}$$

denoting the energy consumed by the k-th load, its estimate in [kWh] and the normalized absolute distance (NAD). The gain in [dB] in Table 1 is calculated as the ratio of NAD before ($\hat{x}_k(n) = x(n)$) and after single load extraction.

5. CONCLUSIONS

A combination of HMM and DNN is useful to extract single loads from aggregate power signals in NILM systems. This new approach is eventless and suitable for variable loads (using a large number of states). It outperforms FHMM because there is no need of knowledge about the remaining loads in the aggregate signal except for the target load. By using multiple DNN-HMM networks, multiple loads can be extracted from the same aggregate signal. The extracted state sequences can be used for semantic analysis of the loads in the next step. The results achieved for the low frequency (1/3Hz) REDD dataset are promising for a low cost NILM system.

6. REFERENCES

[1] G. W. Hart, "Nonintrusive appliance load monitoring," *Proc. IEEE*, vol. 80, pp. 1870–1891, 1992.

[2] M. Zohrer and F. Pernkopf, "Representation models in single channel source separation," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, April 2015, pp. 713–717.

[3] J. Le Roux, J.R. Hershey, and F. Weninger, "Deep nmf for speech separation," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, April 2015, pp. 66–70.

[4] Yuanwei Jin, E. Tebekaemi, M. Berges, and L. Soibelman, "Robust adaptive event detection in non-intrusive load monitoring for energy aware smart facilities," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, May 2011, pp. 4340–4343.

[5] M. Zeifman and K. Roth, "Nonintrusive appliance load monitoring: Review and outlook," *IEEE Trans. Consumer Electronics*, vol. 57, pp. 76–84, 2011.

[6] J. Froehlich, E. Larson, et al., "Disaggregated end-use energy sensing for the smart grid," *IEEE Trans. Pervasive Computing*, vol. 10, pp. 28–39, 2011.

[7] A. Zoha, A. Gluhak, et al., "Nonintrusive load monitoring approaches for disaggregated energy sensing: A survey," *Sensors*, vol. 12, pp. 16838–16866, 2012.

[8] K. S. Barsim, R. Streubel, and B. Yang, "An approach for unsupervised non-intrusive load monitoring of residential appliances," in *2. NILM Workshop*, 2014.

[9] A. Marchiori, D. Hakkarinen, et al., "Circuit-level load monitoring for household energy management," *IEEE Trans. Pervasive Computing*, pp. 40–48, 2011.

[10] M. J. Johnson and A. S. Willsky, "Bayesian nonparametric hidden semi-Markov models," *J. Machine Learning Research*, pp. 673–701, 2013.

[11] M.J. Reyes-Gomez, B. Raj, and D.R.W. Ellis, "Multichannel source separation by factorial hmms," in *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*, April 2003, vol. 1, pp. I–664–I–667 vol.1.

[12] M. Baranski and J. Voss, "Genetic algorithm for pattern detection in NIALM systems," in *Proc. of IEEE Int. Conf. on Systems, Man and Cybernetics*, 2004, pp. 3462–3468.

[13] K. Suzuki, S. Inagaki, et al., "Nonintrusive appliance load monitoring based on integer programming," in *Proc. SICE Annual Conf.*, 2008, pp. 2742–2747.

[14] T. Zia, D. Bruckner, and A.Zaidi, "A hidden Markov model based procedure for identifying household electric loads," in *Proc. IECON*, 2011, pp. 3218–3223.

[15] J. Z. Kolter and M. J. Johnson, "REDD: A public data set for energy disaggregation research," in *Proc. of SustKDD workshop on Data Mining Applications in Sustainability*, 2011.

[16] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, Sanjoy Dasgupta and David Mcallester, Eds. May 2013, vol. 28, pp. 1139–1147, JMLR Workshop and Conference Proceedings.

[17] Pierre Baldi and Peter J Sadowski, "Understanding dropout," in *Advances in Neural Information Processing Systems 26*, C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, Eds., pp. 2814–2822. Curran Associates, Inc., 2013.

[18] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio, "Theano: a CPU and GPU math expression compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010, Oral Presentation.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.