The Neural Energy Decoder: Energy Disaggregation by Combining Binary Subcomponents

Henning Lange
Department of Civil and Environmental Engineering
Carnegie Mellon University
Pittsburgh, USA
Email: henningl@andrew.cmu.edu

Mario Bergés
Department of Civil and Environmental Engineering
Carnegie Mellon University
Pittsburgh, USA
Email: marioberges@cmu.edu

***Abstract*—In this paper a novel approach for energy disaggregation is introduced that identifies additive sub-components of the power signal in an unsupervised way from high-frequency measurements of current. In a subsequent step, these sub-components are combined to create appliance power traces. Once the sub-components that constitute an appliance are identified, energy disaggregation can be viewed as non-linear filtering of the current signal. The approach introduced here tries to avoid numerous pitfalls of existing energy disaggregation techniques such as computational complexity issues, data transmission limitations and prior knowledge of appliances. We test the approach on a publicly available dataset and report an overall disaggregation error of 0.07.**

I. Introduction

Energy disaggregation, the problem of inferring the power consumption of appliances given voltage and current readings at a limited number of sensing points in a building, has received increasing attention in recent years [7, 8]. Energy disaggregation techniques based on latent variable models, such as Factorial Hidden Markov Models (FHMM), have been employed recently with some success [3, 5], and have quickly become a popular approach to the problem. However, even after considering a number of simplifying assumptions, exact inference for FHMMs is intractable and approximate inference techniques are still computationally expensive.

More recently, deep learning has been employed to the problem of energy disaggregation [4], after having made substantial improvements in other fields such as computer vision, speech recognition and machine translation. For a general introduction to artificial neural networks and deep learning, see [6]. Energy disaggregation approaches based on deep learning so far have tried to directly infer the power consumption of individual appliances given the aggregate signal. However, this type of approaches may not generalize well given the difficulties of characterizing the effects that any other appliance (beyond the one of interest) may have on the aggregate signal.

The approach introduced here tries to overcome some of these problems by, first, training a deep neural network to identify subcomponents in the aggregate signal and by, then, combining these subcomponents into appliances. Binary subcomponents are identified from high frequency current readings in an unsupervised way by training a neural network whose activation in the one but last layer is binary (0 or 1) and linear in the output layer. The output of the binary node denotes whether or not the corresponding subcomponent is active. Inferring whether or not an appliance is active then boils down to identifying the relations between subcomponents and the state of the appliances. After training, inferring the state of the appliances can be seen as applying a non-linear filter to high frequency power readings. In section II the method of identifying subcomponents of the aggregate signal is explained. Section III shows how these subcomponents can be used to infer whether or not appliances are turned on at any given time. In section IV it is shown how the proposed algorithm was applied to the BLUED dataset whereas section V concludes the work and proposes future work.

II. Subcomponent identification

As explained earlier, binary subcomponents are identified using a deep neural network. For this, the stream of measured current readings is sliced according to zero-crossings detected in the voltage signal. This results in a matrix $I \in \mathbb{R}^{C \times T}$ with T being the number detected cycles and C the number of samples within one cycle (i.e. 200 for a sampling rate of 12kHz and a line frequency of 60Hz). Let $i(t)$ with $t \in [0, T]$ be the current measured within cycle t (i.e., the t-th column of I). A seemingly trivial mapping is learned from one cycle of current measurements to the active and reactive power consumed within that cycle. Let the active and reactive power be $P(t)$ and $Q(t)$, respectively. $P(t)$ and $Q(t)$ can be computed from the current and voltage data so in a sense the function that the network will learn is already known[1]. However, the topology of the network is constrained in such a way that the network has to piece the aggregate signal together using a limited number of additive components. This is achieved by using a linear activation in the output layer and a binary activation

[1]Though the voltage is not directly represented in I, the cycles in each column of I are aligned with voltage zero-crossings.

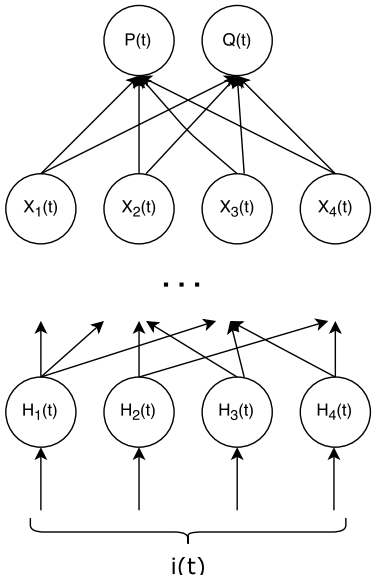


Fig. 1. A graphical representation of the neural network used to identify binary additive subcomponents. x_i denote nodes with binary outputs. h_i denote hidden units. P and Q denote units whose target is active and reactive power respectively.

in the one but last layer. Figure 1 shows a graphical depiction of such a network. Let $x(t) \in (0,1)^N$ be the activation of the binary layer given the input $i(t)$ with N being the hyper-parameter that controls the number of inferred sub-components, and $w_l \in R^{N\times 2}$ be the weights connecting the binary and output layer. Since the active power consumed by an appliance cannot be negative, w_l is constrained in such a way that its first column must be greater or equal to 0. The model can be trained in a fully unsupervised way: given a cycle of current measurements $i(t)$, the desired output, namely active and reactive power can be computed from current and voltage data and the network is then trained to reconstruct the active and reactive power.

In order to minimize its training error, the network is forced to activate some of its binary units and adjust the binary-linear weights in such a way that it can capture most of the variability of the aggregate signal.

The activations of the hidden units in layers lower than the one but last are unconstrained. In principle any activation can be used. In this work a leaky rectifying linear unit is employed whose activation is: $f_r(x) = max(ax, x)$. The activation of the binary units is defined as:

$$f_b(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{else} \end{cases}$$

The network is trained using stochastic gradient descent which requires that the activation functions be differentiable. In order to alleviate the problem of *vanishing gradients*, activation functions are desired whose derivative is mostly non-zero (wide working range), however the derivative of f_b is always 0 except for $x = 0$ where it is not defined. Thus, the binary activation can be understood as

$$f_b(x) = \lim_{a\to\infty} \frac{1}{1+e^{-ax}}$$

Let $s(x) = (1+e^{-ax})^{-1}$. Then the derivative of $s(x)$ asymptotically tends to 0 for $\lim_{x\to\pm\infty}$ and the bigger a, the faster $s(x)$ becomes practically 0 for digital computing purposes. In order to avoid *vanishing gradients*, the gradient of $f_b(x)$ is set to $\frac{ds(x)}{dx}$ with $a = 1$.

III. Combining binary subcomponents into appliances

Once the network is trained, it can be used to extract a binary matrix from high frequency current readings. Let this matrix be $X \in (0,1)^{T\times N}$ with N being the number of components and T the number of current and voltage cycles. The jth row of X denotes the hidden states of the network $x(j)$ given input $i(j)$. We also assume knowledge of a matrix containing ground truth of whether or not a specific appliance is turned on or off at any given voltage cycle. Let $G \in (0,1)^{T\times A}$ denote this matrix with A being the number of appliances. Note that this assumes that all appliances are two-state appliances, i.e. they can either be *on* or *off*.

The additive subcomponents that the network identifies are not unique if the observed signal does not contain all combinations of appliance states. For example, imagine there are two devices a_1 and a_2 which consume $200W$ and $250W$ respectively. Assume also that a_1 and a_2 are never active at the same time. The values that the aggregate signal can take are therefore $0W$, $200W$ and $250W$. A neural network with two binary units has two choices to capture all the variability of the aggregate signal: The weights connecting the binary and the output layer can either be $w_l = [200W, 250W]$ or $w_l = [200W, 50W]$ [2].

Let g_1 and g_2 be boolean variables that denotes whether or not a_1 and a_2 are active respectively. Also, let x_1 and x_2 be the boolean variable representing the binary hidden layer of the network. In case one, inferring g_i from x_i is trivial, since $g_i = x_i$. In the second case however, a non-linear relation between g_i and x_i holds: $g_1 = x_1 \wedge \neg x_2$ and $g_2 = x_1 \wedge x_2$. Following the intuition gained from this simple example, two methods to combine the binary components into appliances will be introduced.

A. Greedy Search

The first method that can be used to aggregate binary subcomponents into a time series representing the activity of appliances is based on a greedy search over boolean functions. Let $x_i \in (0,1)^T$ be the boolean time series that represents the ith binary component, i.e. the ith column of X. Similarly, let $g_i \in (0,1)^T$ represent the ith column of G, i.e. the binary time series representing whether or not

[2] $w_l = [250W, -50W]$ is excluded because of the constraint indicating that weights coming into the 'active power'-node have to be greater than 0

appliance i is turned on. Finally, let $\wedge$ denote the element-wise boolean and-operator such that $x_i \wedge x_j \in (0,1)^T$. A function $f(x_j, ..., x_k)$ is sought that maximizes the similarity between g_i and the output of that function on a subset of binary components $x_j, ...x_k$. We furthermore assume that f is of the type $f(x_j, ...x_k) = a_i x_i \wedge ... \wedge a_k x_k$ with a_i being able to negate x_i. As a similarity measure the F1 score is used.
A greedy algorithm that iteratively adds one component to the function is used. The algorithm is initialized believing that g_i is always turned on, i.e. $f_0 = (1)^T$. Then it iterates over all x_i and computes the F1 score that would result in either appending $\wedge x_i$ or $\wedge \neg x_i$ to f. After one sweep the component that resulted in a maximal increase in F1 score is selected and appended to f. This is repeated until the F1 score cannot be improved further.

B. Logistic regression

The Greedy Search algorithm is only able to find a function of a very simple type. Thus, a more general approach that can model a binary response given predictors is desirable. Logistic regression can be used to predict the probability of binary probabilistic outcomes. We are interested in modeling $P(g_i(t) = 1|x(t))$. Using logistic regression to model this distribution assumes that $P(g_i(t) = 1|x(t))$ is a Bernoulli distribution with

$$P(g_i(t) = 1|x(t)) = \frac{1}{1 + exp[w_i^T x(t)]}$$

The model parameter w_i can be obtained using a maximum likelihood estimate.

C. Naïve Energy Estimation

A naïve approach to estimate the energy consumption of individual appliances can be employed. The power in the *off* and *on* state of each appliance is assumed to be constant and known. Let p_{on} and p_{off} be the power consumption of the appliance and $\hat{g}$ its predicted binary activation vector. For the predicted power trace then the following holds: $\hat{p} = p_{on}\hat{g} + p_{off}(1 - \hat{g})$.

IV. Experiments

The performance of the proposed algorithm is tested on Phase B of the BLUED dataset [1]. For this, zero-crossings were detected in the voltage signal to detect cycles. The number of data points within one cycle varies between 199 and 201 (12kHz sampling rate with a line frequency of $\sim 60Hz$). If a cycle contained less than 201 data points, the data was zero-padded which resulted in a matrix $I \in \mathbb{R}^{T \times 201}$. The dataset was temporally randomized (the rows were shuffled) and the min and max values of every column were extracted and each column was normalized into a range of $[-1, 1]$. The active and reactive power of every column was computed using Budeanu's definition of reactive power [2]. The real and imaginary portions of the Fourier transform of every row were computed and stored.
A 6-layer neural network was created using the python package *keras*[3]. Its topology was not optimized. The first 4 layers have an output dimensionality of 170 and a leaky rectifying linear activation ($f(x) = max(0.5x, x)$). In order to enforce stability, the weights of these layers are constrained such that their norm cannot exceed 3. The 5th layer has an output dimensionality of 100 and uses the binary activation described earlier. Thus 100 binary subcomponents are identified. The last layer corresponding to the active and reactive power has a linear activation and an output dimensionality of 2 (active and reactive power). The weights are constrained in such a way that the bias must be 0 and the weights coming into the node representing active power must be greater or equal to 0. The network was trained to learn the mapping from the real and imaginary parts of the Discrete Fourier Transform of the current signal to active and reactive power using stochastic gradient descent with 100.000 random data points at a time. Since this step was completely unsupervised, the network was trained on all the data.
After the network was trained, the last layer of the neural network was removed and the network was used to infer the binary activations. The ground truth data containing appliance-level consumption was provided at 1Hz, whereas the binary subcomponent activations are at a sampling rate of 60Hz, therefore the binary subcomponents were downsampled to 1Hz. This was done by extracting the data from within one second and taking the median (majority vote).
The matrix containing the power consumption of appliances was binarized using simple thresholding. The data was then temporally randomized and split into two parts, one for training and one of testing (2-fold cross-validation). The mean power consumption in the *on* and *off* states in the training portion of the data was computed and naïve energy estimation was employed to infer power traces of the individual appliances in the testing portion of the data. As a measure of the performance of the energy disaggregation the *mean disaggregation error* was used:

$$mde(p, \hat{p}) = \sum_{i,t} \frac{|p_i(t) - \hat{p}_i(t)|}{p_i(t)}$$

A. Results

Table I shows the results on the individual appliances for which ground truth data was available. The column "active" shows the proportion at which the appliance is active. Since the energy was estimated assuming 2-state appliances, *mean disaggregation error* captures the variance of the appliance to some degree. A perfect 2-state prediction of an appliance with higher variance will always lead to a higher *mean disaggregation error*, even though the energy in sliding windows (or instantaneous power at a lower temporal resolution) would be predicted perfectly. The lower bound for assuming 2-state appliances and a perfect prediction is shown in the column $MDE(p)$.
Whether or not the algorithm can detect if an appliance is

[3] http://keras.io/

Appliance	Active	Greedy F1	Logit F1	MDE($\hat{p}$)	MDE(p)
A/V LR	60%	0.88	0.98	0.05	0.009
Computer 1	27.3%	0.80	0.95	0.12	0.042
Desk Lamp	21.4%	0.87	0.97	0.07	0.013
DVR	20.7%	-	0.99	0.01	0.005
Socket LR	$> 0.1\%$	0.75	0.97	0.09	0.089
Garage Door	0.4 %	-	0.89	0.07	0.063
Iron	0.1 %	0.72	0.95	0.12	0.115
Laptop 1	33.3%	0.76	0.90	0.39	0.272
LCD Monitor	16.2%	0.76	0.91	0.19	0.029
Monitor 2	17.3%	0.65	0.85	0.30	0.089
Printer	0.1%	-	0.66	0.05	0.045
Tall Desk Lamp	21.4%	0.87	0.97	0.07	0.008
TV Basement	20.7%	0.92	0.99	0.05	0.029
Random	30%	-	0	-	-
Overall				0.07	0.037

TABLE I

"ACTIVE" DENOTES THE PROPORTION THE APPLIANCE WAS ACTIVE, "GREEDY" AND "LOGITSHOW THE PERFORMANCE OF THE GREEDY SEARCH AND LOGISTIC REGRESSION, "MDE($\hat{p}$)" AND "MDE(p)" SHOW MEAN DISAGGREGATION ERROR OF THE INFERRED POWER TRACE (LOGIT) AND THE LOWER BOUND ASSUMING 2-STATE APPLIANCES

turned on or off was measured using the F1 score (0 - worst, 1 - best). As expected combining binary subcomponents using logistic regression outperforms the approach based on greedy search. Using logistic regression, for 8 out of 13 appliances, an F1 score greater than 0.95 was achieved, for 2 out of 13 appliances an F1 score between 0.90 and 0.95 was achieved and for 2 appliances an F1 score between 0.85 and 0.90 was achieved. Only one appliance scored lower than 0.85 (printer with 0.66).

V. CONCLUSION

The Neural Energy Decoder has shown unprecedented results on the BLUED dataset. In order to infer the states and to some degree the power consumption of appliance with reasonable precision only a single cycle of high frequency current data needs to be provided. No other energy disaggregation system is capable of doing this to the knowledge of the authors. On top of that, once the model is trained inference is computationally extremely cheap. The approach can also circumvent data transmission limitation that high frequency energy disaggregation approaches face: an energy disaggregation system that requires a sampling frequency of 12kHz needs to collect and store at least 4GB of data per sensing point per day. Such a system is unlikely to scale to many buildings. The Neural Energy Decoder can alleviate this problem by computing the state of the binary subcomponents on a smart meter and only sending these features out. Since the state of a binary unit can be encoded by a single bit the data to be transmitted can be reduced to 1MB in order to obtain energy disaggregation results with a temporal resolution of 1Hz.

The system also handles unmodeled devices gracefully. Once the binary subcomponents are identified, modeling appliances is decoupled, i.e. in contrast to e.g. latent variable approaches which require to jointly model all appliances, this approach could in principle be used to infer the power consumption of any number of devices without having to explicitly model their dependencies.

VI. FUTURE WORK

Online or stream processing: The algorithm described was used in a fully offline fashion which in turn would require buffering of all the data until inference. In a real life setting, it would be desirable if the unsupervised pre-learning can be carried out in an online way which does not require buffering of the entire dataset. Energy consumption datasets exhibit high temporal structure and temporal randomization was required for effective pre-training. Identifying the binary subcomponents in an online way probably requires to buffer some data points and selecting those data points to be buffered is crucial in making online subcomponent identification work.

Unsupervised appliance models: In order to make the algorithm completely unsupervised, one could try to force the deep neural network to infer binary components such that a simple unsupervised approach can piece the components together. Recall the example from earlier where there are two ways for the network to accommodate all of the variability of a time series containing two appliances which consume 200W and 250W but are never turned on at the same time, namely $w_l = [200W, 250W]$ and $w_l = [200W, 50W]$. The first explanation already disaggregates the energy successfully whereas for the second explanation a non-linear function would have to be learned in an unsupervised way. Since we assume that both appliances are never turned on at the same time, the binary subcomponent matrix for the first explanation is much sparser than for the second explanation. Enforcing sparsity in the binary layer of the matrix can in theory force the network to identify "easier" subcomponents to combine in an unsupervised way.

Exploiting temporal structure: So far, the algorithm makes no use of temporal information. Exploiting temporal information (beyond the temporal structure encountered within every voltage cycle) could improve the performance for some appliances. A straight-forward approach could be to infer the appliance activities using Conditional Random Fields instead of Logistic Regression or to use recurrent deep networks instead of a feed-forward network.

Multi-state appliances: So far, all appliances were modeled as 2-state appliances. Logistic regression can in principle handle multinomial responses (targets), therefore extending the algorithm to handle multi-state appliances is straight-forward and its merit should be investigated.

Energy Estimation: So far, a naïve energy estimator that assumes constant power for the *on* and *off* states of each appliance was used. Every binary subcomponent is associated with active power, i.e. the weight between the corresponding binary unit and the output layer. This information is not used so far. A more fine grained approach to the problem of energy estimation could take the activity and the associated active power of the binary units into account.

REFERENCES

[1] Kyle Anderson et al. "BLUED: A fully labeled public dataset for event-based non-intrusive load monitoring research". In: *Proceedings of the 2nd KDD workshop on data mining applications in sustainability (SustKDD)*. 2012, pp. 1–5.

[2] LS Czarnecki. "Budeanu and Fryze: Two frameworks for interpreting power properties of circuits with nonsinusoidal voltages and currents". In: *Electrical Engineering* 80.6 (1997), pp. 359–367.

[3] Matthew J Johnson and Alan S Willsky. "Bayesian Nonparametric Hidden Semi-Markov Models". In: *The Journal of Machine Learning Research* 14.1 (2013), pp. 673–701.

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

[5] J Zico Kolter and Tommi Jaakkola. "Approximate Inference in additive Factorial HMMs with application to Energy Disaggregation". In: *International conference on artificial intelligence and statistics*. 2012, pp. 1472–1482.

[6] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature* 521.7553 (2015), pp. 436–444.

[7] Michael Zeifman and Kurt Roth. "Nonintrusive Appliance Load Monitoring: Review and outlook". In: *IEEE Transactions on Consumer Electronics* (2011), pp. 76–84.

[8] Ahmed Zoha et al. "Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey". In: *Sensors* 12.12 (2012), pp. 16838–16866.