Greedy Deep Disaggregating Sparse Coding

Shikha Singh
IIIT Delhi
shikhas@iiitd.ac.in

Manoj Gulati
IIIT Delhi
manoj@iiitd.ac.in

Angshul Majumdar
IIIT Delhi
angshul@iiitd.ac.in

***Abstract*— This is the first study combining deep learning with sparse coding (dictionary learning) to solve the energy disaggregation problem. In the first layer, we learn a disaggregating dictionary. In subsequent layers, we learn sparsifying dictionaries. The first layer is learnt from the smart-meter data; subsequent layers are learnt from features obtained from the previous layer. For disaggregation, instead of using a single level of dictionary as the basis for each device, we use the multiple level dictionaries. Experimental results show that by going deeper, it is possible to improve disaggregation accuracy.**

Keywords*— *Energy Disaggregation, Dictionary Learning, Deep Learning,

I. Introduction

The approach towards energy disaggregation is broadly based on the nature of the targeted household and commercial appliances. These appliances can be broadly categorised as simple two-state (on/off) appliances such as electrical toasters and irons; more complex multistate appliances like refrigerators and washing machines; and continuously varying appliances such as IT loads (printers, modems, laptops etc.). The earliest techniques were based on using real and reactive power measured by residential smart meters. The appliances' power consumption patterns were modelled as finite state machines [1]. These techniques were successful for disaggregating simple two state and multistate appliances, but they performed poorly in the case of time-varying appliances which do not show a marked step increase in the power. More recent techniques, based on stochastic finite state machines (Hidden Markov Models) [2], have improved upon the prior approach. In a pioneering work [3], the problem was addressed using dictionary learning. The advantage of dictionary learning is that it does not require the simplifying assumptions that are required by prior techniques; as long as there is enough data, dictionary learning will be able to model the appliance, at least in theory. Later studies like [4] proposed variations of the basic framework proposed in [6] to solve the disaggregation problem.

In the past decade or so, there have been a renewed interest in deep neural networks. While the basic architecture for such machine learning tools were known a priori, practical versions of these arrived only around 2005. With the advent of graphical processing units in everyday computers and heuristics tricks like greedy learning [5], deep neural networks became a implementable approach. Today deep learning is used by all large research corporations as the de facto tool for data analysis.

Around the same time (i.e. 2006) dictionary learning was popularised only by the advent of K-SVD [6]; even though the basic concept was known from much before [7]. Dictionary learning is very popular in academic circle; it has been used to solve problems arising in vision, speech, imaging, reconstruction etc.

Both deep learning and dictionary learning fall under the broader purview of representation learning. However, so far, dictionary learning have been a shallow approach. Also, there have been no study to assimilate the two into a single framework. For the first time in this work, we recast dictionary learning in a neural network framework. This automatically allows us to extend the dictionary learning framework into deeper architectures.

It must be noted that our work is not related to hierarchical / structured dictionary learning techniques [8-10]; although the title of [10] carries the terms 'deep', 'sparse' and 'coding' – it is basically a hierarchical approach; not a deep one. Hierarchical learning is a shallow (single level) learning technique, where a single level of dictionary is learnt, but the dictionary atoms maintain a hierarchical structure. It is similar to 'learning' a wavelet like decomposition for 'tree-structured' sparsity on any piecewise smooth signal.

In this work we discuss the similarity between neural networks and dictionary learning. We build on that to learn a deeper architecture. Each level is then trained with a greedy approach. The deeply trained model is used for disaggregation. In our approach, the first level is trained by disaggregating sparse coding [4], while the subsequent layers is a trained by simple sparse coding. Results show improvement going from 1st to 2nd layer, but saturates in the 2nd layer; there is no improvement in results with the 3rd layer.

II. Dictionary Learning and Neural Nets

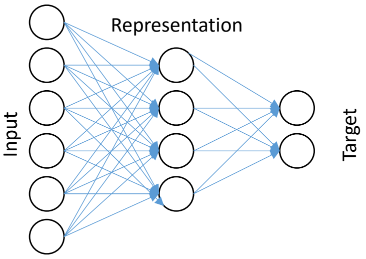

Figure 1. Single Representation Layer Neural Network

The relationship between dictionary learning and neural network has been independently proposed in a recent unpublished work [11]. The idea followed here is the same.

A simple neural network with one representation (hidden) layer consists of two networks (Figure 1) – one between the input and the hidden layer and the other between the hidden layer and the target; the problem is to learn the network weights. Learning the weights between the representation and the target is straightforward. The challenge is to learn the network weights (from input) and the representation. This is the study of representation learning.

Restricted Boltzmann Machine (RBM) is one technique to learn the representation layer. Broadly speaking, RBM learning is based upon maximizing the similarity between the projection of the data and the representation, subject to the usual constraints of probability. Multiple layers of RBM are stacked (loosely speaking) to form a DBN.

RBM has a probabilistic formulation. The other prevalent technique to train the representation layer of a neural network is by autoencoder.

$$\min_{W,W'} \|X - W'\phi(WX)\|_F^2 \qquad (1)$$

The cost function for the autoencoder is expressed above. *W* is the encoder, and *W'* is the decoder. The autoencoder learns the encoder and decoder weights such that the reconstruction error is minimized. Essentially it learns the weights so that the representation $\phi(WX)$ retains all the information of the data, so that it can be reconstructed back. Once the autoencoder is learnt, the decoder portion of the autoencoder is removed and the target is attached after the representation layer. Multiple units of such autoencoders are nested inside the other to form SAE.

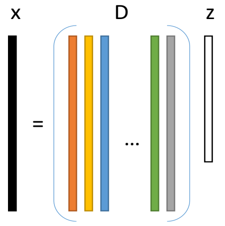

Figure 2. Dictionary Learning

The interpretation for dictionary learning is different. It learns a basis (*D*) for representing the data (*X*). The columns of *D* are called 'atoms'. In this work, we look at dictionary learning in a different manner. Instead of interpreting the columns as atoms, we can think of them as connections between the input and the representation layer; see Figure 3.

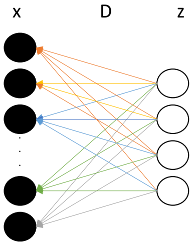

Figure 3. Neural Network type Interpretation

Unlike a neural network which is directed from the input to the representation, the dictionary learning kind of network points in the other direction – from representation to the input. Dictionary learning employs an Euclidean cost function (2), given by

$$\min_{D,Z} \|X - DZ\|_F^2 \qquad (2)$$

This is easily solved using alternating minimization. In every iteration, the first step is to update the coefficients assuming D is fixed and the next step is to update the dictionary assuming the coefficients are fixed. This alternating update of dictionary and coefficients continue till the algorithm converges to some local minima. Today most studies (following K-SVD [6]) impose an additional sparsity constraint on the representation (Z).

III. Proposed Approach

A. Deep Architecture

We have established the connection between dictionary learning and neural network kind of representation learning. Building on that, we propose deeper architecture with dictionary learning. An example of two layer architecture is shown in Figure 4.

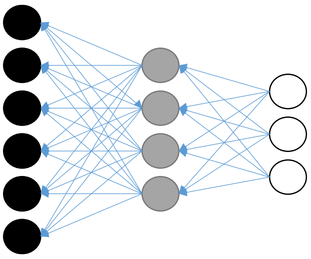

Figure 4. Deep Dictionary Learning

For the first layer, a dictionary is learnt to represent the input data. In the second layer, the representation from the first layer acts as input; it learns a second dictionary to represent the features / representation from first level. This concept can be extended to deeper layers. In this work we follow a greedy paradigm [5] for our deep dictionary learning problem.

B. Discriminating Layer

Disaggregating Discriminative Sparse Coding (DDSC) [3] is a complex technique. Given the limitations of sparse, we cannot describe it in detail. We just explain the final formulation.

$$\min_{D_1^{(i)}\text{'s}, Z_1^{(i)}\text{'s}} \sum_{i=1}^{N} (\|X^{(i)} - D^{(i)}\,\mathrm{Z}_1^{(i)}\|_F^2 + \lambda\|Z_1\|_1$$
$$s.t. \mathrm{Z}_1^{(i=1:N)} = \arg\min_{Z\geq 0} \|X - D_1 Z\|_F^2 + \lambda\|Z\|_1 \qquad (3)$$

Here $X = \sum_i X^{(i)}$ is the aggregated signal, $\left[Z_1^{(1)T} | \ldots | Z_1^{(N)T}\right]^T = Z_1$ (first level of coefficients), $\left[Z^{(1)T} | \ldots | Z^{(N)T}\right]^T = Z$ and $D_1 = \left[D_1^{(1)} | \ldots | D_1^{(N)}\right]$ (first level of dictionaries). The superscript (i) denotes the ith device;

there are *N* such devices. The subscript denotes 1st layer of dictionary learning.

The dictionaries $D^{(i)}$'s and coefficients $Z^{(i)}$'s are learnt by the initial sparse coding (SC) step:

$$\min_{D^{(i)},Z^{(i)}} \left\|X^{(i)} - D^{(i)}Z^{(i)}\right\|_F^2 + \lambda\left\|Z^{(i)}\right\|_1 \quad (4)$$

In (3) $D_1^{(i)}$'s are discriminatively optimized bases such that it can produce activations $Z_1^{(i)}$'s which are as close as possible to the activations that can produce minimum disaggregation error. Activations that can produce minimum disaggregation error is obtained by,

$$\min_{Z_1} \left\|X - D_1Z_1\right\|_F^2 + \lambda\left\|Z_1\right\|_1 \quad (5)$$

The activation functions cannot be negative, so in each iterations a non-negativity constrained on applied on the *Z* (all values less than zero are imputed to zero).

C. Deeper Layers

The first layer gives us a disaggregating basis that yields discriminative representation. Since we are learning the layers in a greedy fashion, there is no feedback from a subsequent layer to a previous layer. Therefore the discrimination happens only in the first layer. In subsequent layers, we want to learn basis for more abstract representation for each appliance.

Given the representation from the first layer, we learn the second layer using simple sparse coding, i.e. by solving,

$$\min_{D_2^{(i)}Z_2^{(i)}} \left\|Z_1^{(i)} - D_2^{(i)}Z_2^{(i)}\right\|_F^2 + \lambda\left\|Z_2^{(i)}\right\|_1 \quad (6)$$

This can be solved using alternating minimization. There are theoretical guarantees behind convergence of this algorithm [13]. In order to prevent a degenerate solution where *D* is very large and *Z* is very small, the columns of *D* are normalized after every iteration.

The same procedure can be continued for deeper layers. One can take the representation from the previous layer as the input and learn a basis and a deeper level of representation from it. However, as seen in deep learning, one cannot keep on going deeper and expect better and better results. Deeper layers mean more parameters to train, given the limited amount of data, the issue of over-fitting arises. This leads to saturation of results after a few layers and starts degrading if one goes deeper.

D. Disaggregation

For the disaggregation phase, the total (aggregate) power read by the meter is assumed to be a sum of the power consumed by individual devices. This is expressed as:

$$X = \sum_i X^{(i)} = [D^{(1)} | \ldots | D^{(N)}]\begin{bmatrix} Z^{(1)} \\ \ldots \\ Z^{(N)} \end{bmatrix} \quad (7)$$

where $D^{(\mathrm{i})} = D_1^{(i)}D_2^{(i)}D_3^{(i)}\ldots$, i.e. the basis for representing each appliance is the combined deep basis.

The idea is the same as [3], but instead of having a single layer of learnt basis, we have multiple layers. The sparse representation is learnt by solving the l_1-norm minimization problem.

$$\min_Z \left\|X - DZ\right\|_F^2 + \lambda\left\|Z\right\|_1 \quad (8)$$

where $D = [D^{(1)} | \ldots | D^{(N)}]$ and $\mathrm{Z} = [Z^{(1)T} | \ldots | Z^{(N)T}]^T$.

Once the loading coefficients are estimated, the consumption for each appliance is obtained by:

$$\hat{X}_i = D_i Z_i,\ i = 1 \ldots \mathrm{N} \quad (9)$$

IV. Experimental Results

We conduct this experiment on a subset of Dataport dataset available in NILMTK (non-intrusive load monitoring toolkit) format, which contains 1 minute circuit level and building level electricity data from 240 houses. The data set contains per minute readings from 18 different appliances: air conditioner, kitchen appliances, electric vehicle, and electric hot tub heater, electric water heating appliance, dish washer, spin dryer, freezer, furnace, microwave, oven, electric pool heater, refrigerator, sockets, electric stove, waste disposal unit, security alarm and washer dryer. We are assigning 165 homes for training and 72 for testing. The remaining 3 homes do not have aggregated data so they are not used in the experiments.

To prepare training and testing data, aggregated and sub-metered data are averaged over a time period of 10 minutes. This is the usual protocol to carry out experiments on the Pecan street dataset. Each training sample contains power consumed by a particular device in one day while each testing sample contains total power consumed in one day in particular house. The evaluation metric has also been defined in the previous work as 'disaggregation accuracy'.

$$Acc = 1 - \frac{\sum_t \sum_n \left|\hat{y}_t^{(i)} - y_t^{(i)}\right|}{2\sum_t \bar{y}_t}$$

where *t* denotes time instant and *n* denotes a device; the 2 factor in the denominator is to discount the fact that the absolute value will "double count" errors.

Table. 1. Energy Disaggregation Accuracy (in %)

H.No.	Sparse Coding (SC)	1st Layer (DDSC)	2nd Layer (SC)	3rd Layer (SC)
1.	94.92	94.04	95.69	95.64
2.	64.75	68.50	71.34	71.65
3.	83.60	83.31	84.92	84.91
4.	43.67	47.68	51.33	51.01
5.	63.21	66.05	68.81	68.78
6.	43.23	51.24	56.66	56.53
7.	58.89	64.81	70.31	70.33
8.	51.41	57.32	59.81	60.10
9.	58.39	62.10	65.40	65.38
10.	62.96	64.85	66.00	66.43
11.	50.37	57.65	62.20	62.26
12.	58.13	64.09	64.78	64.98
13.	51.97	56.25	58.32	58.03
14.	78.26	79.03	78.81	78.83
15.	64.06	65.84	64.11	64.34
16.	68.09	77.23	82.83	82.83
17.	63.34	75.38	78.71	79.24
18.	67.29	72.44	71.40	71.77

19.	69.05	82.63	87.84	87.88
20.	91.86	92.76	94.67	94.72
21.	48.50	56.16	61.79	61.55
22.	51.27	55.55	57.92	57.88
23.	69.00	76.82	78.28	79.04
24.	85.10	88.75	90.63	90.61
25.	59.04	58.05	57.93	57.88
26.	91.20	90.57	92.25	92.22
27.	69.48	70.16	74.37	74.48
28.	82.27	82.21	83.55	83.59
29.	65.07	70.76	74.08	74.48
30.	50.99	61.53	66.37	65.96
31.	55.60	61.08	62.47	62.63
32.	47.91	56.49	60.77	60.94
33.	50.87	54.80	61.41	61.41
34.	74.76	76.74	77.42	77.43
35.	60.88	64.60	66.46	67.11
36.	54.36	59.09	63.07	63.51
37.	56.55	54.72	53.33	54.10
38.	62.08	66.38	69.34	69.46
39.	83.42	85.32	86.96	86.97
40.	52.68	57.62	56.40	56.65
41.	86.15	87.18	88.50	88.49
42.	87.06	89.93	89.15	89.14
43.	77.27	78.07	77.64	77.86
44.	56.66	59.36	62.25	62.06
45.	57.49	61.46	67.18	67.55
46.	58.38	62.68	62.27	62.58
47.	47.76	52.99	54.11	53.80
48.	80.05	84.03	88.13	88.09
49.	78.36	86.52	87.32	87.29
50.	49.82	49.83	50.89	51.29
51.	48.71	51.85	51.50	51.24
52.	45.77	47.40	52.37	52.20
53.	78.53	77.88	79.42	79.41
54.	52.31	49.58	52.05	52.09
55.	40.52	50.97	52.50	52.89
56.	53.69	63.37	64.39	64.68
57.	66.31	69.59	72.58	73.09
58.	70.41	76.71	77.25	77.58
59.	75.50	76.18	79.13	78.96
60.	53.59	59.77	59.46	59.57
61.	47.13	50.19	52.29	51.98
62.	52.54	55.48	55.90	56.24
63.	53.27	57.96	59.17	59.32
64.	74.97	81.65	83.15	83.56
65.	50.99	54.26	55.02	54.70
66.	46.88	52.89	57.02	57.26
67.	58.14	63.68	64.41	64.33
68.	73.87	81.74	82.61	82.95
69.	57.67	57.83	59.13	59.25
70.	79.91	83.64	84.88	85.45
71.	47.82	54.06	56.12	56.32
72.	79.47	82.37	85.87	86.11
Total	**63.13**	**67.25**	**69.34**	**69.45**

The detailed experimental results are shown in Table 1. The sparse coding technique refers to the simple solution proposed in [4] (4); where dictionaries for sparsely coding each device are learnt. DDSC is the final approach proposed in [4] (3); it learns dictionaries to generate discriminative sparse coefficients. We use the DDSC as the first layer. In subsequent layers (2 and 3) we use sparse coding. The first level (both for DDSC – column 2 and SC – column 1) uses 300 dictionary atoms. The second level uses 70 atoms and the third level uses 30 atoms. These number of atoms were decided on a smaller validation set (REDD).

DDSC improves upon SC by about 4 percent. We see that going deeper, we can improve the results by another 2 percent.

V. Conclusion

In this study we represent dictionary learning / sparse coding and deep learning in a common framework of neural network type representation. This allows us to propose a new area called 'deep sparse coding' – we learn multiple layers of dictionaries to sparsely encode the data. We follow the greedy paradigm for training.

This work specifically studies the problem of energy disaggregation. So far, shallow (single layer) dictionary learning techniques were being used for addressing the problem. We take the time-tested DDSC method (3) to learn the first layer of dictionaries. In the subsequent layers, we use simple sparse coding. Going deeper, allows us to improve upon the DDSC technique by about 2% in terms of disaggregation accuracy.

VI. Acknowledgement

Authors acknowledge the support provided by ITRA project, funded by DEITy, Government of India, under grant with Ref. No. ITRA/15(57)/Mobile/HumanSense/01.

References

[1] H. G. Wells, "Nonintrusive appliance load monitoring", Proceedings of the IEEE, Vol. 80, pp. 1870-1891, 1992.

[2] H. Kim, M. Marwah, M. Arlitt, G. Lyon, and J. Han, "Unsupervised Disaggregation of Low Frequency Power Measurements," in SIAM Conference on Data Mining, pp. 747-758, 2011.

[3] Z. Kolter, S. Batra, and A. Y. Ng., "Energy Disaggregation via Discriminative Sparse Coding," NIPS, 2011.

[4] E. Elhamifar and S. Sastry, "Energy Disaggregation via Learning 'Powerlets' and Sparse Coding", AAAI 2015.

[5] Y. Bengio, "Learning deep architectures for AI", Foundations and Trends in Machine Learning, Vol. 1(2), pp. 1-127, 2009.

[6] M. Aharon, M. Elad and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation", IEEE Transactions on Signal Processing, Vol. 54 (11), pp. 4311-4322, 2006.

[7] K. Engan, S. Aase, and J. Hakon-Husoy, "Method of optimal directions for frame design," IEEE ICASSP, 1999.

[8] Y. Suo, M. Dao, U. Srinivas, V. Monga, T. D. Tran, "Structured Dictionary Learning for Classification", arXiv:1406.1943

[9] L. Bar and G. Sapiro, "Hierarchical dictionary learning for invariant classification", IEEE ICASSP, pp. 3578 – 3581, 2010.

[10] H. Dong, B. Wang, C.-T. Lu, "Deep Sparse Coding Based Recursive Disaggregation Model for Water Conservation", IJCAI, 2013.

[11] S. Tariyal, A. Majumdar, R. Singh, M. Vatsa, "Greedy Deep Dictionary Learning", arXiv:1602.00203.

[12] I. Daubechies, M. Defrise, C. De Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint", Communications on Pure and Applied Mathematics, Vol. 57: 1413-1457, 2004.

[13] A. Agarwal, A. Anandkumar, P. Jain and P. Netrapalli, "Learning Sparsely Used Overcomplete Dictionaries via Alternating Minimization", COLT, 2014.